(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
5 April 2001 (05.04.2001)

PCT

(10) International Publication Number
WO 01/24045 A2

(51) International Patent Classification⁷: G06F 17/30

(21) International Application Number: PCT/CA00/01042

(22) International Filing Date:
8 September 2000 (08.09.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/407,336 29 September 1999 (29.09.1999) US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 09/407,336 (CIP)
Filed on 29 September 1999 (29.09.1999)

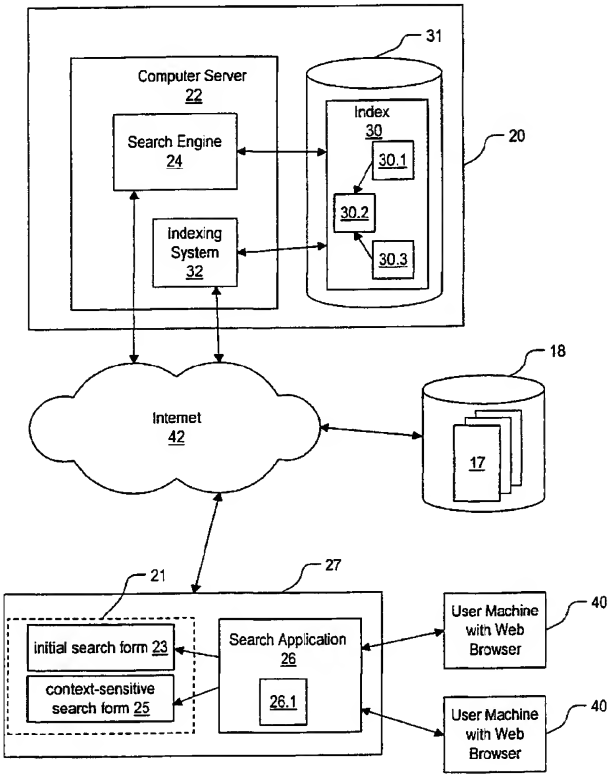
(71) Applicant *(for all designated States except US)*: **XML-GLOBAL TECHNOLOGIES, INC.** [US/US]; 701 Dexter Avenue, Suite 420, Seattle, WA 98109 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **MACKENZIE, Chad, Matthew** [CA/CA]; 1238 Richards Street, Suite 1806, Vancouver, British Columbia V6B 3G2 (CA). **CANNON, Finlay** [CA/CA]; 969 West 10th Avenue, Vancouver, British Columbia, V5Z 1L9 (CA). **NICKULL, Duane, Allan** [CA/CA]; 199-1027 Davie Street, Vancouver, British Columbia V6E 4L2 (CA). **HOGLUND, Jamie, Michael, Thomas** [US/CA]; 1516 Davie Street, Suite 1108, Vancouver, British Columbia V6B 2V6 (CA).

(74) Agents: **SUM, Theodore, W.** et al.; Smart & Biggar, Box 11560, Vancouver Centre, Suite 2200, 650 West Georgia Street, Vancouver, British Columbia V6B 4N8 (CA).

[Continued on next page]

(54) Title: METHOD, SYSTEM, SIGNALS AND MEDIA FOR INDEXING, SEARCHING AND RETRIEVING DATA BASED ON CONTEXT



(57) Abstract: A search engine receives an initial set of search criteria from a user and proceeds to initiate a search of a database containing an index of documents and their contents. The search engine searches the database for the location of documents containing data that matches the initial search criteria, and contextual terms (if any) that are recorded in the database as providing context to one or more of the data elements within any of the documents retrieved. A list of identified documents and a list of contextual terms are generated and transmitted to the user for display in a context-sensitive search form. The context-sensitive search form provides a refinement mechanism for enabling the user to refine the list of identified documents by selecting one or more of the cotextual terms from the list of contextual terms presented. When the search engine receives one or more of the contextual terms from the user, a refined list of documents is generated and transmitted to the user for further selection.

WO 01/24045 A2

(81) **Designated States** *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) **Designated States** *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

METHOD, SYSTEM, SIGNALS AND MEDIA FOR INDEXING, SEARCHING AND RETRIEVING DATA BASED ON CONTEXT

This application is a continuation-in-part of, and claims the benefit of, U.S. Patent Application Serial No. 09/407,336 filed September 29, 1999.

FIELD

The present invention relates generally to a method, system, signals and media for indexing, searching and retrieving data based on context.

BACKGROUND

The Internet has rapidly become one of the leading communications mediums of our age. One of the most popular applications used in the Internet is the World Wide Web. Tens of thousands of Web sites around the world house hundreds of millions of Web pages and related structured documents.

Yet as the volume and diversity of information available over the Internet continues to grow, the ability to locate relevant information is becoming increasingly more challenging. This challenge is complicated by the fact that much of the structured data available over the Internet generally lacks any centralized organization beyond the organization of Web pages amongst particular Web sites. As a result, it is increasingly difficult for users to quickly and easily locate the information that they need. In this ever growing networked environment, search engines have become an important tool for enabling users to search for and retrieve information that is relevant to their needs. Popular search engines for searching the Internet are available from Yahoo™ (http://www.yahoo.com), Infoseek™ (http://www.infoseek.com), Lycos™ (http://lycos.cs.cmu.edu), AltaVista™ (http://www.altavista.com), Excite™ (http://www.excite.com), Microsoft Network™ (http://www.msn.com), HotBot™ (http://www.hotbot.com), Northern Light™ (http://www.nlsearch.com) and others.

Several conventional search engine services organize the information contained within their databases into broad categories. Yahoo™, for instance, provides users with a number of categories within which they may narrow their search including such categories as "government", "entertainment", and "health". Users are able to browse these categories in order to narrow their search to structured documents indexed within a particular category. While these broad categories provide users with some mechanism for organizing the nature of their search, they do not enable a user to perform a search for particular data which has been given specific context within structured documents.

The Extensible Markup Language (XML) provides one solution to the need to include context sensitive data within structured documents accessible over the Internet and other networks. XML was introduced in part to serve as the basis for applications that permit Web authors and publishers to create XML-based Web pages containing structured context sensitive data. While XML is being rapidly adopted by organizations, existing search interfaces that are difficult to understand and use, and provide limited functionality.

In order to enhance the searching of XML documents and other structured documents having context sensitive data, it would be desirable to have a simple yet effective tool for performing context-based searching. It would be further desirable to have a context sensitive search query interface that organizes a search for context sensitive data in an easy to understand manner.

SUMMARY OF THE INVENTION

The above and related desires are addressed in the present invention by providing a method, system, signals and media for indexing, searching and retrieving data based on context. The present invention can be applied to index, search and retrieve context sensitive data associated with structured documents or structured data created with the Extensible Markup Language

(XML), an XML-derived markup language, or another context sensitive markup language. The present invention can also be applied to indexing, searching and retrieving other types of context sensitive data.

In accordance with one aspect of the invention, a computer-implemented method is provided for retrieving data based on context. In this aspect, a search criterion is received from a requesting party and used to find a set of data sources containing a data element that matches the search criterion. For structured documents retrieved as part of the set of data sources, a set of structural components is retrieved that provide context to the data element found in the retrieved structured documents. The set of structural components and references to the set of data sources are transmitted to the requesting party for further processing.

In accordance with another aspect of the invention, a computer-implemented method of retrieving data is provided. In this aspect, a set of structural components is identified based on one or more search criteria received from a requesting party. The set of structural components are transmitted to the requesting party for selection. A selected structural component is received and references are retrieved to structured documents that contain data associated with the selected structural component. These references are then transmitted to the requesting party. The set of structural components may be identified as a set of contexts associated with a set of structured documents. A set of references may be identified to structured documents that contain data marked by at least one of the set of structural components. These latter references may be transmitted to the requesting party with the transmission of the set of structural components for selection. In another variation, the set of structural components identified based on the one or more search criteria may be identified as being associated with a set of document structures.

In accordance with another aspect of the invention, there is provided a computer-implemented method of indexing data into a database. In this aspect, a data source is indexed within the database. The data source is scanned in search of structural components that provide context to any data elements within the data source. Such data elements and their associated structural components are retrieved from the data source. Organizational information representing an organization of the data elements and the structural components within the data source is also retrieved and stored within the database.

Other aspects and features of the present invention will become apparent to those ordinarily skilled in the art upon review of the following description of specific embodiments of the invention in conjunction with the accompanying figures.

BRIEF DESCRIPTION OF THE DRAWINGS

In drawings which illustrate embodiments of the invention,

FIG. **1** is a block diagram of a system for indexing, searching, and retrieving data according to a first embodiment of the invention;

FIG. **2** is a block diagram illustrating further aspects of the system shown in FIG. **1**;

FIG. **3** is a block diagram illustrating the indexing system shown in FIG. **1**;

FIG. **4** is a flow diagram illustrating the submission of a data source for indexing by the indexing system in FIG. **3**;

FIG. **5** is a flow diagram illustrating the indexing of data sources by the indexing system in FIG. **3**;

FIG. **6** is a diagram illustrating a sample XML document that can be indexed in the system in FIG. **1**;

FIG. **7** is a block diagram illustrating a sample content submission request for the submission of an XML document to the indexing system in FIG. **3**;

FIG. **8** is a block diagram illustrating a sample submission response issued by the indexing system in FIG. **3**;

FIG. **9** is a diagram illustrating a textual components layer for the sample XML document in FIG. **6**;

FIG. **10** is a diagram illustrating a structural components layer for the sample XML document in FIG. **6**;

FIG. **11** is a block diagram illustrating a data structure for textual components retrieved by the system in FIG. **3** from the sample XML document in FIG. **6**;

FIG. **12** is a block diagram illustrating a data structure for structural components retrieved by the system in FIG. **3** from the sample XML document in FIG. **6**;

FIG. **13** is a diagram illustrating a sample entry in a computer file to store data regarding unique document terms encountered by the system in FIG. **3** during the indexing process;

FIG. **14** is a block diagram illustrating a structure for a node within the index used in the first embodiment;

FIG. **15** is a block diagram illustrating a structure of a postings blob for a textual component managed by a node in accordance with the first embodiment;

FIG. **16** is a block diagram illustrating a structure of a postings blob for a structural component managed by a node in accordance with the first embodiment;

FIG. **17** is a block diagram illustrating an arrangement of key/blob pairs in accordance with the present invention;

FIG. **18** is a block diagram illustrating an alternative arrangement of the index of the first embodiment, including a computer-readable file for channel mappings;

FIG. **19** & **20** are flow diagrams illustrating a method of searching for data based on context according to the first embodiment of the invention;

FIG. **20A** is a flow diagram illustrating a portion of FIG. **19** according to the first embodiment of the invention;

FIG. **21** is a block diagram illustrating the sequence of information exchanged as data signals between a user machine and a search engine via an intermediary search application in accordance with the first embodiment of the invention;

FIG. **22** is a diagram illustrating the structure of an initial search form presented to the user in accordance with the first embodiment in FIG. **1**;

FIG. **23** is a diagram illustrating the structure of a first representation of a context-sensitive search form presented to the user in accordance with the first embodiment in FIG. **1**;

FIG. **24** is a diagram illustrating the structure of a second representation of the context-sensitive search form presented to the user in accordance with the first embodiment in FIG. **1**;

FIG. **25** is a diagram illustrating a sample raw XML search request in accordance with the first embodiment in FIG. **1**;

FIG. **26** is a diagram illustrating a sample raw XML search response in accordance with the first embodiment in FIG. **1**;and

FIG. **27** is a block diagram of a system for searching, and retrieving data according to another embodiment of the invention;

DETAILED DESCRIPTION

Reference will now be made in detail to implementations and embodiments of the invention, examples of which are illustrated in the accompanying drawings.

In one aspect of the invention, there is provided a context-sensitive search query interface for searching and retrieving data based on context. In another aspect of the invention, there is provided an indexing system for indexing data elements and their data sources based on the context of such data elements in their data sources.

FIG. **1** is a block diagram illustrating a system **20** for providing search engine services according to a first embodiment of the invention. The system **20** has at least one computer server **22**, a search engine **24** running on the computer server **22**, and a database **31** which contains an index **30** to indexed data including, but not necessarily limited to, context sensitive data. The computer server **22** is provided with conventional communications equipment for communicating over a network or internetwork such as the Internet **42**.

In the first embodiment, the search engine **24** is programmed to conduct user-initiated searches of the index **30** to retrieve references to structured documents **17**, and other data sources including, but not limited to, other structured data sources, which are cataloged within the index **30**. A user communicates with the search engine **24** via a web browser (or micro-browser or other navigational tool) running on one of the user machines **40**. Examples of commercially available web browsers include Netscape Navigator, Microsoft Internet Explorer, and Mosaic. An intermediary search application **26** runs on a web computer server **27** supporting user-based searching of the index **30**. The display of the search interface (i.e. the search forms) on a user's web browser, including the display of the search results, and communications with the search engine **24**, are handled by the intermediary search application **26** which serves as an interface between users and the search engine **24**.

The user interacts with the search engine **24** via search forms **21** displayed by the user's web browser. The search forms **21** are stored on the intermediary search application **26** as computer-readable instructions. The search forms **21** provide an end-user with a sequence of search forms to assist the user in defining and refining the user's search for relevant information within the index **30**. With the search forms **21**, users can define search criteria using well-known search definition techniques such as keyword searches and phrase searches. The search engine **24** searches the index **30** for relevant data based on the search criteria that the user has defined via a search form.

The user initiates a search of the index **30** by retrieving an initial search form **23** from the intermediary search application **26**. This may be performed by simply connecting one of the user machines **40** to a Web site running on the web computer server **27** and serving as the human interface to the search engine system **20**. The user machines **40** can be any type of computing device capable of communicating with the intermediary search application **26** or the search engine **24**. In the first embodiment, the user machines **40** are personal computers connected to the Internet. Other types of user machines may also be used, for example wireless hand-held computing devices and other microprocessor-based electronic devices having a web browser and a network connection. In the first embodiment, the user machines **40** can access the services of the search engine **24** through a dial-up connection with an ISP and a network connection established over the Internet **42**. Other connections by the user machines **40** may be established. For example, the user machines **40** may access the search engine services through an intranetwork, a cable or xDSL modem connection, a wireless connection or dedicated network connection (e.g. LAN or WAN) or the like.

When the initial search form **23** is displayed to a user on one of the user machines **40**, the user is prompted to provide one or more search criteria in order to proceed with a search. The search criteria generated by a user using the initial search form **23** serves as a search request that is transmitted as a set of variables via intermediary search application **26** to the computer server **22** for further processing by the search engine **24**. When a search request is received by the computer server **22**, the search request is used by the search engine **24** to define and initiate a search of the index **30**. The search results generated from this search are sent by the search engine **24** via the intermediary search application **26** to the user's web browser where the results are displayed using a context-sensitive search form.

The first embodiment provides at least two types of search forms which can be communicated to the user machines **40**: initial search form **23**, and a context-sensitive search form **25**. Examples of search forms **21** are shown in FIG. **22** to **24** which are discussed later in this specification. Referring to FIG. **1**, the search forms **21** are implemented in the first embodiment using HTML and are displayed as Web pages on the user machines **40**. It will be appreciated by persons skilled in the art, however, that other programming techniques can be used (independently, in combination or in addition) to encode the search forms in place of or in addition to HTML. For instance, XML or another SGML-based markup language may be used. As another example, applets may be used such as Java Servlets, Server Side Includes, Java™, Javascript™, ActiveX™, or Active Server Pages™ or other equivalent computer-readable instructions that can be invoked either locally on a user machine or via command over a network.

<u>User-based searching</u>

The intermediary search application **26** acts as an intermediary between the user's web browser and the search engine **24**. The intermediary search application **26** receives and converts user-based search requests into raw XML-based search requests which it transmits to the search engine **24**. The intermediary search application **26** also receives and converts raw XML search results from the search engine **24** into a user-presentable format which the intermediary search application **26** transmits to the user's machine. Web server software **28** runs on the web computer server **27** to support the intermediary search application **26**. The web server software **28** can be any of several well-known server packages including, for example, Apache's web server software or Microsoft's Internet Information Server.

The intermediary search application **26** comprises a plurality of CGI scripts in the first embodiment. When a user formulates a set of one or more search criteria search via a search form and uses the search form to request that a search be conducted based on the search criteria, an HTTP message (Hypertext Transport Protocol message) is transmitted from the user's machine (**40**) to the web computer server **27**. The HTTP message sent from the user serves as a user-based search request containing an Internet Protocol (IP) address associated with the web computer server **27**, the name of a common gateway interface (CGI) script **26.1** residing on the web computer server, and query parameters for configuring the CGI script **26.1** with the formulated search criteria. The web server software **28** launches the CGI script **26.1** with the transmitted query parameters. The CGI script **26.1** converts the format of the user's formulated search criteria into a raw XML search request which the web computer server **27** transmits to the search engine **24** where the search of the index **30** is performed, and waits for a response. Once the search results are retrieved by the search engine **24**, they are transmitted as a raw XML response to the web computer server **27** where the CGI script **26.1** converts the raw XML response into a format that is presentable to the user's browser. In this embodiment, the CGI script **26.1** provides the actual display information to the user's browser for the display of search forms and search results.

Indexing

The index **30** is a storage structure that is used to catalog data and the location of the source(s) of such data, including, but not limited to, context sensitive data and the location of the source(s) of such context sensitive data. In the first embodiment, the sources of data include, amongst other types of data, structured documents **17** located locally or on an accessible network resource **18**. The structured documents **17** contain data elements which are marked with structural components that provide context to such data elements. Data elements that are marked by (or surrounded by) such structural components represent context sensitive data. The structural components are represented by contextual markup tags based on a markup language. Data elements which may be marked with contextual markup tags include character data and other markup tags (for example, graphical or multimedia objects). The term "character data" refers to textual components of the structured documents **17** which are not part of a markup tag contained within the structured documents **17**. The textual components of a document are made up of one or more characters based on a binary-coded character set containing letters, numbers and other typographic symbols. Preferably, the textual components are Unicode compliant. The Unicode standard is a universal character encoding standard used for the representation of text for computer processing. The Unicode standard conforms with ISO/IEC 10646 and is supported by the Unicode Consortium (http://www.unicode.org). Other examples of binary-coded character sets which may be used include ASCII (American Standard Code for Information Interchange), EBCDIC (Extended Binary Coded Decimal Interexchange Code), and BCD (Binary Coded Decimal).

Examples of contexts which may be used in one or more structural components to provide context to data elements are illustrated in Table **1** below. Nested contexts are shown separated by slashes "/".

Name
Name/First Name
Name/Last Name
Person/Name/Last Name
Address
Address/PostalCode
Address/Country
Inventory
Part/SKU
Part/PartNumber
Inventory/Vehicles/Transmission/Part/PartNumber
Play
Play/Act
Play/Act/Line
Email
Email/Sender
Email/Recipient
Email/Message
Email/Message/Reply
Email/Mime_Headers/X-Originating-IP

Table 1

In the first embodiment, the structured documents **17** are XML documents. Other types of structured documents and structured data may also be processed using the present invention such as, for example, data formatted with an XML-derived markup language or another context-sensitive markup language, as well as other forms of data containing context sensitive data. Quasi-structured documents may also be processed, such as HTML documents, WML documents and standardized text formats such as those employed by academics (eg. essay or dissertation formats).

Data sources such as the structured documents **17** are cataloged (or indexed) within the index **30** as part of an index building and maintaining process supported by indexing system **32**. The indexing system **32** is implemented as a set of computer-readable instructions running on the computer server **22** or another computer having access to index **30**.

Referring to FIG. **2**, the computer server **22** has at least one processor **46** (a central processing unit in the first embodiment) connected via a bus **43** to a computer-readable medium **45**. The computer-readable medium **45** provides a memory store for software and data residing on the computer server **22**. The computer-readable medium **45** can include one or more types of computer-readable media including volatile memory such as Random Access Memory (RAM), and non-volatile memory such as a hard disk or Read Only Memory (ROM). In the first embodiment, RAM **48** and a hard disk drive **49** each serve as computer-readable media.

The search engine **24** resides on a hard disk drive **49** and is loaded via bus **43** into RAM **48** as computer-readable instructions which execute on the processor **46** to provide search engine services to user machines **40** directly or indirectly connected to the system **20**. The search engine **24** runs as an application on an operating system **47**. The indexing system **32** also runs on the operating system **47**. The operating system can be any of several well-known operating systems such as, for example, Windows NT^{TM}, Windows **2000**^{TM}, MacOS^{TM}, UNIX, Linux or the like. For the first embodiment, the operating system is Linux. The computer server **22** includes a network interface **44** which is in communication with the processor **46** to connect the computer server **22** to a network so that the computer server **22** can communicate with user machines **40** or other networked devices.

Referring to FIG. **3**, the indexing system **32** comprises a content submission interface **50**, a content acceptor **52**, a queue **54** and an indexer **56**. The data sources, such as the structured documents **17** in FIG. **1**, are submitted by a submitting party (for example, a user or application) to the content submission interface **50** which runs as a computer program on the computer server **22** (FIG. **1**) or another computer server having access to the index **30**. The content submission interface **50** monitors for content submission requests. A content submission request can be received from either a user or an application. A content submission request contains a data source intended for insertion into the index **30**. Context sensitive data is submitted in a content submission request in the form of a structured document or in the form of a structured data stream which preserves the context of such context sensitive data. A content submission request also contains resource location information identifying the location of a data source on a resource. In the first embodiment, the resource location information is represented by a Uniform Resource Locator (commonly known as a URL) which specifies the location of the data source (for example, one of structured documents **17**). For the purpose of this specification, the term "resource" refers to any computer-implemented object or data that can be accessed via a computer network (e.g. LAN, WAN, etc.) or internetwork such as the Internet, and which contains or refers to a data source. Examples of resources include Web sites, Web pages, file directories, URIs, URNs, URLs, IP addresses, POP, Email (MIME or S/MIME), electronic data files and other electronic documents accessible over a network.

The resource location information is a form of metadata that identifies an attribute of a data source. Other metadata may also be included within the content submission request to further specify attributes of a data source to be indexed within index **30**.

In the first embodiment, the content submission interface **50** is implemented using sockets. When a request is made to a socket from a remote host, the content submission interface **50** accepts the request, gives the remote host a connection and processes the content submission request sent to the content submission interface **50** over the socket.

Referring to FIG. **3** and **4**, a content submission request received via the content submission interface **50** is scanned by the content acceptor **52** which is programmed to determine if the content submission request contains content in an acceptable format. The content of an accepted content submission request is placed on queue **54** for subsequent indexing into index **30**. The content acceptor **52** informs the submitting party whether or not the content of the content submission request has been accepted for indexing.

In the first embodiment, content submission requests contain XML documents which contain data elements that are given context using markup tags. An example of an XML document is shown in FIG. **6** generally at **60** containing data elements **62** marked by contextual markup tags **64** (structural components) which contain contextual terms that provide context to marked data elements **62**. Attributes may also be used within the XML document to provide context to data. Attribute names are equivalent to markup tags **64** and attribute data is equivalent to data elements **62**.

FIG. **7** shows a sample of a raw content submission request **66** received from a submitting party via the content submission interface **50** for processing by the content acceptor **52** in FIG. **3**. The content submission request **66** contains a request to submit the XML document shown in FIG. **6** for indexing within index **30**, a copy of the XML document **60** and metadata **68** regarding the submitted XML document. The metadata **68** includes resource location information **65** which is a URL identifying the location of the original XML document **60**. Preferably, the metadata **68** also includes other information regarding a submitted document, including, for example, channel information **67**, the type of document (MIME type **69**), the date the document was last modified, and a title and abstract for the document. Channel information **67** is used to index a document under a channel within the index **30**. A document may be indexed into several channels within index **30**. With channels, a user or application can search the entire index **30** or a subset of the index **30** under one or more channels. For instance, using channels a search may be narrowed by document type.

Referring to FIG. **3** and **7**, the content acceptor **52** scans the content submission request **66** to determine if the submitted content is in an acceptable format. Preferably, when scanning the content submission request, the content acceptor **52** checks the metadata **68** for the type of document that has been submitted. If the document is in XML, then the content acceptor checks the remainder of the content submission request **66** to, for instance, check that the document is in well-formed XML and to check whether or not the document is already in the index **30**. If the document is identified as something other than an XML document, the content acceptor **52** invokes a document handler **58** to transform the document into XML. Context sensitive data within the submitted document is preserved during the conversion process. If the submitted document cannot be transformed into XML, the content acceptor **52** will not accept the document. Preferably, all submitted documents are processed by a version of the document handler, with documents that are not in XML being transformed into XML, and with XML documents not undergoing such transformation. A document that is not in XML is transformed into XML by a version of the document handler programmed to recognize the submitted document type and transform it into XML.

If the content acceptor **52** accepts the content submission request **66** then a submission response is transmitted to the submitting party confirming acceptance. An example of content submission response is shown in FIG. **8** at **70** acknowledging that the content submission request in FIG. **7** has been accepted and the submitted document and meta-data concerning the document have been queued for indexing. If the content acceptor **52** rejects the content submission request **66**, then a submission response is transmitted to the submitting party notifying the submitting party of the rejection.

Referring to FIG. **3** and **5**, content, for instance structured content such as a structured document and its metadata, which has been queued for indexing is processed by the indexer **56**. The indexer **56** runs as a computer program on the computer server **22** or another computer server having access to the index **30**. The structured content in the first embodiment is represented by XML documents and their metadata residing in queue **54** awaiting processing. The indexer **56** retrieves an XML document and its metadata from the queue **54**, parses the document and its metadata, and then catalogs the parsed information within the index **30**.

The indexer **56** parses the submitted XML document in order to retrieve and temporarily store data elements from the document and structural components within the document that provide context to any of the retrieved data elements. Preferably, the retrieved data elements and structural components are stored in data structures to maintain the association between such structural components and corresponding data elements, and to record information representing the organization of the data elements and the structural components within the document that is being parsed.

During parsing, in order to record the organization of data elements and structural components, a submitted XML document is preferably divided into two logical layers: a textual components layer and a structural components layer. The textual components layer contains textual components representing any words and any other non-markup character sequences retrieved from the submitted XML document, and any other data elements that are marked in the submitted XML document with contextual markup tags. The structural components layer contains structural components retrieved from the submitted XML document, including contextual terms from contextual markup tags that provide context to data elements in the submitted XML document. If a submitted document contains no structural components, then the structural components layer is empty.

FIG. **8** and **9** illustrate a textual components layer **72** and a structural components layer **74** respectively for the sample document in FIG. **6**. The textual components layer **72** is shown containing the words found in the sample document **60**. The structural components layer **74** is shown containing contextual terms **76** from contextual markup tags used in the sample document **60** to provide textual components with context.

Logically dividing the textual components and the structural components into different layers allows the indexer **56** (FIG. **3**) to separately manage the indexing of textual components and structural components parsed from the submitted XML document. The indexer **56** keeps track of position information and nesting information for both textual components and structural components. In addition, the indexer **56** keeps track of boundary information identifying which textual components are first and last surrounded by the structural components. With the layered approach to parsing the submitted XML document, position information and nesting information for textual components are stored in a data structure, while position information, nesting information and boundary information for structural components are stored in another data structure.

FIG. **11** shows, by way of example, a data structure **80** for textual components retrieved from the sample XML document **60** in FIG. **6**. The data structure **80** is used by the indexer **56** in FIG. **3** to store the position information **82** and nesting information **84** for the words and other non-markup character sequences retrieved from the sample XML document **60**. A particular piece of position information in data structure **80** represents the position of a particular textual component relative to the other textual components retrieved from the XML document. The nesting information **84** represents the depth at which a corresponding textual component occurs inside a structural component. Thus if a textual component has a structural component with a depth of 3, the nesting information for the textual component would be 4.

FIG. **12** shows, by way of example, a data structure **86** for structural components retrieved from the sample XML document **60**. The data structure **86** is used by the indexer **56** in FIG. **3** to store position information **87**, nesting information **88** and boundary information **89** for the structural components retrieved from the sample XML document **60**. The position information **87** is used to identify the hierarchical relationship between structural components. The position information **87** for data structure **86** identifies the position of a structural component within the sample XML document **60** relative to the other structural components. The nesting information **88** for data structure **86** is used to identify whether any structural components are nested within other structural components, and if so, the depth of the nesting. The depth of the nesting is relative to the root structural component. The boundary information **89** stored as "begin" and "end" fields in the data structure **86** is used to identify, for any structural component retrieved, the position information in the data structure **80** of the first and last textual components surrounded by such structural component in the sample XML document **60**.

Other types of data structures may be used to record the information shown in FIG. **11** and **12** and are considered equivalent.

Referring to FIG. **3**, a computer-readable file, referred to in the first embodiment as TermList file **30.1**, is used to store data regarding each of the unique document terms that the indexer **56** encounters when parsing submitted documents. Both textual components and structural components retrieved from a parsed document are treated by the indexer **56** as document terms. The TermList file **30.1** serves as a translation table between actual document terms and numeric identifiers that index structures within the index **30** use to refer to the actual document terms.

As an XML document is parsed by the indexer **56**, each retrieved document term that is unique to the TermList file **30.1**, whether the document term is a textual component or a structural component, is assigned a unique identifier number which, in combination with the corresponding unique document term, are stored as part of a new entry in the TermList file **30.1**. If the retrieved document term already has an existing entry in the TermList file **30.1**, then the existing entry is updated to modify frequency information about the retrieved document term.

A sample entry in the TermList file **30.1** is illustrated in FIG. **13** generally at **90**. The entry **90** contains a variable ("term_string") to store the actual string representation of a document term retrieved from a parsed document. Another variable ("term_id") is used to store a unique numeric identifier assigned to the retrieved document term. The entry **90** also preferably stores statistical information about the total number of instances that a document term appears in the index and the total number of documents that contain such document term. This statistical information is used by the search engine **24** (FIG. **1**) to prioritize queries on the index **30** and to determine weights for document terms that are retrieved as part of search results. In the first embodiment, the entry contains a plurality of variables to store the statistical information. Such statistical information preferably includes information on: the number of times a document term appears as a textual component, the number of times the document term appears as a structural component, the number of documents that contain the document term as a textual component, and the number of documents that contain the document term as a structural component.

Referring to FIG. **3**, once a submitted XML document is parsed and the parsed information is temporarily stored in data structures (such as the data structures in FIG. **11** and **12**), the parsed information is then added by the indexer to an index file **30.2** within index **30** so that the information can be later searched. In the first embodiment, the index file **30.2** is a computer-readable file organized in a B-plus tree structure containing a plurality of nodes having key/value pairs. Data is stored in a blob (or block) in the value chunk of a node. Each blob is structured to store data on either a textual component or a structural component, and a flag in the key of the node identifies whether the data stored in the blob is for a textual component or a structural component.

An illustration of the structure of a node **92** within index file **30.2** is shown in FIG. **14**. Each node **92** has a key portion **94** identifying a particular document term for which the node **92** is being used to store data, and a blob portion **96** for storing organizational data about the particular document term with respect to those data sources from which the particular document term has been retrieved and indexed within index **30**. The blob portion **96** forms the value chunk of the node **92**.

In the first embodiment, the key portion **94** of the node **92** contains an identifier **93** for uniquely identifying the document term for which the node **92** is being used to store data, and a flag **95** which indicates whether the node **92** is storing information for a textual component or a structural component.

Organizational data within the blob portion **96** is stored as a set of postings. Each posting records data identifying position information and nesting information for each instance of the document term in a particular data source. Each posting is associated with a particular data source which has been indexed by indexer **56**.

FIG. **15** illustrates the structure of a postings blob **100** for a textual component that is being tracked by a particular node within the index file **30.2** (FIG. **3**). The postings blob **100** contains a set of postings **102** each of which contains data identifying the position and nesting information (**82** and **84** in FIG. **10**) for each instance of the textual component retrieved from a particular data source. For example, if node **92** is used to store data about the textual component "Introduction" and the sample document in FIG. **6** is indexed, then a posting would appear in the set of postings **102** identifying the document (for example, doc_id1) and the position and nesting information for the textual component "Introduction" in the document.

FIG. **16** illustrates the structure of a postings blob **104** for a structural component that is being tracked by a particular node within the index file **30.2** (FIG. **3**). The postings blob **100** contains a set of postings **102** each of which is used to store the position information, nesting information and boundary information (**87**, **88** and **89** in FIG. **11**) for each instance of the structural component retrieved from a particular data source.

Referring to FIG. **17**, preferably the key/blob pairs are structured so that a pointer from a key points to its corresponding blob portion so that blobs are not necessarily stored in contiguous chunks in the index file **30.2** (FIG. **3**).

As illustrated in FIG. **14**, the key portion **94** of a node **92** preferably includes a low document ID indicator **97**. The low document ID indicator **97** indicates the lowest document ID contained within a particular blob. The low document ID indicator **97** can be used during a search for multiple terms within the index **30** to speed up the retrieval of the results. If, during the search for a first term, the search engine **24** finds that the first term only occurs in documents numbered, for example, 1001, 1007 and 1024, the search for the second term could ignore any node having a low document ID of less than 1001 or a low document ID of greater than 1024, since such nodes have already been eliminated as possible matches. It will be noted that the low document ID, while providing enhanced retrieval capabilities, is not required to retrieve results.

The low document ID indicator **97** is included in nodes to improve the performance of searches on the index **30**. The lowest document ID within a blob can also be determined during a search by looking inside a node at the contents of its blob portion.

Once an existing node grows to a maximum manageable size, the existing node is split into two nodes. For a very common document term, as with textual components "the" or "and", such a document term will have several nodes that their index information is spread amongst. Alternatively, a new node may be inserted with a low document ID of the first new posting entered in the new node and all subsequent new postings for the document term managed by the existing node will be stored in the new node.

Referring to FIG. **1**, a computer-readable file, referred to here as DocumentList file **30.3** is used to store all meta-data about each data source indexed within index **30**. For each data source indexed within the index **30**, the DocumentList file **30.3** contains an entry recording meta-data about the data source, including, for example, resource location information (**65**), MIME type (**69**), date last modified, title, summary/abstract, size, owner etc. This information can be retrieved as part of the search results obtained by the search engine **24** during a search of the index **30**.

Processing the textual components and structural components of a document in separate layers and indexing such textual components and structural components using the node structure of index **30** allows the index **30** to be more manageable and scalable. As the number of documents indexed within index **30** grows, the index **30** will grow more linearly than would be the case with a relational database. Storing textual components, structural components and their associations with textual components and other structural components in a relational database, the database would grow much larger and complex as the number of documents submitted for indexing increased. In addition, with such a relational database structure, it can be more difficult, or require more processing, to find nestings of terms that are multiple levels apart. With index **30**, ancestor-child relationships between stored structural components can be easily and rapidly retrieved. Moreover, the index **30** stores enough information about textual components and structural components retrieved from documents to enable the search engine **24** to retrieve data based on context, to retrieve nested context, and to retrieve data based on nested context, but at the same time the index **30** is not tied to a specific schema or structure for a document. As illustrated in this first embodiment of the invention, such retrieval can be achieved by indexing within the node structure of index **30** the position information and nesting information for textual components in nodes specific to such textual components, and by indexing position, nesting and boundary information for structural components such as contextual terms in nodes specific to such structural components.

Thus, while a relational database may be used in place of the index **30** for other aspects of the present invention, indexing data based on context from index **30** provides for an efficient configuration which readily supports rapidly retrieving data based on context.

Context Sensitive Search Queries and Interface

FIG. **19** and **20** show flow diagrams illustrating a method of retrieving data based on context by searching for data sources using context sensitive search queries according to the first embodiment of the invention. As a specific example, the flow diagrams in FIG. **19** and **20** will be described with reference to the system **20** shown in FIG. **1**.

In operation, the intermediary search application **26** receives a request to send the initial search form **23** to a requesting party which, in the first embodiment, is one of the user machines **40**. In response, the intermediary search application **26** sends computer-readable instructions representing the initial search form **23** to the requesting user machine **40** where the initial search form **23** is displayed on the local web browser. As illustrated at **220** in FIG. **21**, the computer-readable instructions for the initial search form **23** are transmitted to the requesting user machine **40** as a computer data signal embodied in a carrier wave.

FIG. **22** illustrates the initial search form **23** that is displayed to a user via a web browser **41** running on one of the user machines **40** (FIG. **1**). The initial search form **23** provides an easy and simple mechanism for prompting the user to specify an initial set of search criteria without necessarily having to define the context for the search criteria. As the initial set of search criteria for the first embodiment, a user inputs into one of the user machines **40** one or more search terms via the initial search form **23**. Preferably, the user may also include operators, such as Boolean operators (eg. AND, OR, NOT etc.), to further define the nature of the search desired.

Referring to FIG. **1**, the initial set of search criteria is transmitted as part of a user-based search request to the intermediary search application **26** which converts the user-based search request into a raw XML search request which is then transmitted to the search engine **24** to initiate an initial search of the records of the index **30**. In the first embodiment, the user-based search request is an HTTP message identifying CGI script **26.1** that is part of the intermediary search application **26**, and query parameters for configuring the CGI script **26.1** with the initial set of search criteria. The CGI script **26.1** is launched with the query parameters and converts the format of the initial set of search criteria into the raw XML search request which the web computer server transmits as a computer data signal to the search engine **24**. FIG. **21** illustrates at **222** the transmission of the initial set of search criteria to the intermediary search application and, at **224**, the transmission of the initial set of search criteria as a raw XML search request to the search engine **24**, each as a computer data signal embodied in a carrier wave.

FIG. **25** shows a sample raw XML search request **240** containing sample search criteria **242**. In the sample shown, a basic search request is included for the phrase "off to see the wizard".

Referring to FIG. **1** and **19**, the search engine **24** receives the initial set of search criteria in the raw XML search request at block **200**. Once the search engine **24** receives the initial set of search criteria, the search engine **24** proceeds at block **202** to initiate an initial search of the index **30** in search of matches (or "hits") using the parameters within the initial set of search criteria. A "match" or "hit" represents an entry in the index **30** identifying a data source having data fitting within the parameters of a given set of search criteria. In the first embodiment, the search engine **24** searches the index file **30.2** for nodes which contain document terms that fit within the constraints of the initial set of search criteria.

At block **204**, the search engine **24** organizes an initial set of search results based on the matches found in the index **30**. The search engine **24** retrieves from the index **30** a set of resource location identifiers that identify data sources containing terms, phrases or other data that satisfy the initial set of search criteria. In the first embodiment, the resource location identifiers are Uniform Resource Locators (URLs) which are retrieved and arranged in a list of URLs. If any of the search criteria is identified in the index **30** as being marked with one or more contextual terms in any of the data sources in the list of URLs, then the search results will also include a list of such contextual terms. In one alternative, the search results may also include reference to all unique contextual terms within retrieved data sources, whether or not any of the search terms of the search criteria are marked with such contextual terms in those data sources.

The contextual terms returned in a search can be used to refine the search for relevant data within the index **30** by presenting the requesting party with contextual information (ie. the contextual terms) which the requesting party can use to filter the search results. In an alternative embodiment, the contextual terms may also be used to refine the search by predicting possible document types and presenting them to the requesting party. For instance, if the document type "Invoice" is known in the index **30** to have the structural components "Name", "Address", "Invoice Number", and "ItemNumber", the search engine **24** predicts that the presence of these structural components indicates that certain document types, including the "Invoice" document type, are available to be searched upon within the index **30**. This information can then be communicated to the requesting party so that a particular document type may be used to refine a search for relevant data within the index **30**.

FIG. **20A** illustrates the search and retrieval process performed by the search engine **24** in FIG. **1** and in blocks **202** and **204** of FIG. **19**. When a search is received, the search engine **24** uses the TermList file **30.1** to resolve search terms into numeric identifiers (IDs) at block **300**. Search terms which cannot be mapped to numeric IDs with the TermList file **30.1** are preferably ignored in the remainder of the search. At block **302**, the search engine **24** then searches for the search terms from amongst the nodes in the index file **30.2** and retrieves at block **304** a list of document identifiers that satisfy the search criteria.

In performing a search at block **302**, the search engine **24** is able to perform a wide range of searching for relevant documents based on the position and nesting information for textual components and the position, nesting and boundary information for structural components stored in the index structure of index file **30.2**. For instance, certain search criteria may require that the search engine **24** retrieve documents having textual components side-by-side or in a certain proximity to one another, such as phrase or proximity searching. With the position information of structural components stored within the search engine **24** is able to rapidly determine which documents have terms that fit within such a phrase or proximity search. With the index structure of index file **30.2**, the search engine **24** is also able to rapidly retrieve nesting information and boundary information for a contextual term retrieved during a search without having to traverse during runtime each ancestor of the contextual term. Other search criteria, such as the contextual criteria discussed further below, may require that the search engine **24** retrieve structural components having a particular nesting. Another search criteria may require that the search engine **24** retrieve document identifiers for documents having textual components associated with a certain nesting of structural components. For example, in an alternative embodiment the search criteria may specify that the search engine **24** retrieve a list of documents having the textual component "Francis" within the nested context "Person/Name/Last Name".

Since the index file **30.2** contains the position and nesting information for textual components and the position, nesting and boundary information for structural components, all indexed in advance from source documents, the search engine **24** is able to search the tree structure of the index file **30.2** and rapidly identify relevant search results without having to calculate at runtime the location of textual components within structural components or the nesting levels of structural components. In searching for structural components, the search engine **24** can restrict its searching to nodes in the index file **30.2** that are used to store information on structural components. Similarly, in searching for textual components, the search engine **24** can restrict its searching to nodes in the index file **30.2** that are used to store information on textual components. With searches for textual components associated with a particular context term, the search engine **24** retrieves the position information for the textual component in question and determines if it falls within the boundary information recorded in the index file **30.2** for the particular context term of a document. As another example of the flexibility provided with the index **30**, in order to determine whether a first contextual term (structural component) is a parent or ancestor of a second contextual term (structural component) in a document, the search engine **24** compares the nesting information and boundary information of the two contextual terms. The first contextual term is a parent or ancestor of the second contextual term if the second contextual term has a depth (see FIG. **12** as an example) of greater than the first contextual term, and the begin field (in the boundary information) for the second contextual term is greater than or equal to the begin field for the first contextual term, and the end field (in the boundary information) for the second contextual term is less than or equal to the end field for the first contextual term. The nesting information can also be used to readily determine the number of levels of nesting between a structural component and one or more of its ancestors.

If a channel was specified in the search, the search engine **24** retrieves at block **306** a list of document identifiers from the channel mappings table **30.4** (FIG. **18)** for a list of documents that are associated with the one or more channels specified in the search. In this case, the search engine **24** intersects the list of document identifiers retrieved from channel mappings table **30.4** with the list of document identifiers retrieved from the index file **30.2** to generate search results that contain a list of document identifiers that satisfy the search criteria and that appear in the channel(s) specified.

At block **308** the search engine **24** sorts and ranks the resulting list of document identifiers. At block **310** the search engine **24** looks up the metadata in the DocumentList file **30.3** for those documents identified in the resulting list which are to be presented to the user. The set of retrieved metadata and sorted listed of document identifiers to be presented to the user are then transmitted.

Referring to FIG. **1** and **19**, at block **206** the search engine **24** transmits the initial set of search results as a raw XML response. The raw XML response is transmitted to the intermediary search application running on the web server computer, where the CGI script **26.1** converts the raw XML response into a format presentable to the user's web browser. FIG. **21** illustrates at **226** the transmission of the raw XML response to the intermediary search application as a computer data signal embodied in a carrier wave.

Referring to FIG. **26**, a sample raw XML response **246** containing a sample set of search results is shown generated from a search based on the sample XML search request in FIG. **25**. In the sample illustrated, the response **246** contains a set of hits which identify matching documents, their URLs and summary information regarding each matching document including title, abstract, data last modified, size, rank and score of the document. The summary information returned for each matching document allows the intermediary search application **26** to use a variety of techniques to display or further filter the search results for the user. The response **246** also contains the list of contextual terms **248** found to provide context to search terms within the retrieved documents. The response **246** also contains metadata **247** summarizing the search including the search criteria that formed the basis of the search. Providing metadata **247** about the search criteria in the response **246** allows the intermediary search application to be stateless. As well, with the metadata **247** the search engine **24** can inform the search application **26** of ignored text and structural components such as with fields **250**, and unused text and structural components such as with fields **252**. The one or more fields **250** for ignored text and structural components contains search terms that the requesting party specified to exclude from the scope of a search as well as search terms which the search engine **24** ignored to improve the speed of a search, for instance very common terms such as "and", "the", "or", and "a". The one or more fields **252** for the unused text and structural components contains search terms not used in the search because they did not appear within the index **30**.

Referring to FIG. **1**, the intermediary search application **26** processes the raw XML response with CGI script **26.1** to generate a representation of the context-sensitive search form **25** containing the list of URLs and the list of contextual terms received in the raw XML response. The representation of the context-sensitive search form **25** is transmitted to the user's browser for display. FIG. **21** illustrates at **228** the transmission to a user machine **40** of a computer data signal embodied in a carrier wave, the computer data signal containing computer-readable instructions for the display of the representation of the context-sensitive search form **25** with the list of contextual terms and the list of URLs.

FIG. **23** illustrates, by way of example, the context-sensitive search form **25** presented to the user for the first embodiment. The context-sensitive search form **25** is displayed on a user machine **40** within web browser **41** as a Web page that includes a display area **126** for displaying all or a portion of the list of URLs generated in the initial search. Preferably, each URL is displayed as a hyperlink and includes some information from the corresponding data source (e.g. title, a subset of text retrieved from a document). Data sources referenced by the URLs are accessed by selecting a given hyperlink. In the first embodiment, the context-sensitive search form **25** also includes a contextual display area, which is shown in FIG. **23** as a context box **128**, for displaying all or a portion of the list of contextual terms generated from the initial search of the index **30** (FIG. **1**). The context box **128** and other aspects of the context-sensitive search form **25** are displayed to the user via the web browser in the first embodiment using a meta-language such as HTML or another SGML-based language. It will be appreciated by one skilled in the art that selecting an item, such as a contextual term, from the context box **128** or from another part of the first context-sensitive search form **25** can be achieved using one of many known techniques used for selecting a feature on a web page (or an electronic form) and sending information identifying the selection to another application.

For the context-sensitive search form **25**, the option to view only segments of the list of URLs generated from the initial set of search criteria is helpful, particularly when the list of URLs is quite long. A preferred technique for providing this option is to catalog the search results and to dynamically identify which portions of the search results are to be presented to the user. This may be done by grouping the search results into "pages" or "segments", presenting the user with one of the pages of results, and allowing the user to navigate through the search results on a page-by-page basis. An example of this technique is illustrated with the page links **127** in FIG. **23**. The same technique may be used to allow a user to view the refined search results via the context-sensitive search form **25** in FIG. **24**.

Referring to FIG. **1** and **23**, in addition, the context box **128** preferably appears on each "page" of search results viewed by the user via a search form so that the user can quickly refine search results using contextual terms retrieved from the index **30**.

The context-sensitive search form **25** provides a refinement mechanism for enabling a user to refine, via the web browser **41**, the list of URLs by selecting one (or more) of the contextual terms from the context box **128**. The user can initiate this refinement mechanism by selecting one (or more) of the contextual terms in the context box **128**. The selected contextual terms are encoded in a data signal (**230** in FIG. **21**) that is transmitted to the intermediary search application **26** which converts the transmission into a raw

XML search request that is then transmitted as another data signal (**232** in FIG. **21**) to the search engine **24** to initiate further searching of the index **30**.

Referring to FIG. **1** and **19**, when the search engine **24** receives at block **208** instructions in the form of a raw XML search request to refine the list of URLs using one (or more) of the contextual terms from the list of contextual terms presented to the user, the search engine **24** generates a refined list of URLs from the original list of URLs. The refined list of URLs is generated by determining which of the documents referred to in the original list of URLs includes the contextual term(s) selected by the user. In a preferred arrangement, the refined list of URLs does not include URLs from the original list of URLs referring to data sources that do not include one or more of the original search terms marked by the refining contextual term(s). This latter arrangement helps to narrow the search results by identifying only those data sources within which search terms are used in the context(s) selected to refine the initial search results.

When the search engine **24** receives one or more selected contextual terms at block **208**, the initial search results are filtered at block **210** using the one or more contextual terms selected by the user, and a refined list of contextual terms and a refined list of URLs are generated. The selected contextual terms are used by the search engine **24** to filter from the original list of URLs the references to data sources that are not identified in the index **30** as containing one or more the selected contextual terms.

At block **212** the search engine **24** transmits the refined list of URLs as part of a raw XML response (**234** in FIG. **21**) to the intermediary search application **26** which processes the raw XML response to generate a search form with the refined search results in a form presentable to the user. Preferably, the intermediary search application **26** generates a second representation of the context-sensitive search form **25** containing the refined list of URLs. This second representation of the context-sensitive search form **25** is then transmitted by the intermediary search application **26** to the user's machine.

FIG. **21** illustrates at **236** the transmission to a user machine **40** of a computer data signal embodied in a carrier wave, the computer data signal containing computer-readable instructions for the display of the second representation of the context-sensitive search form **25** with the refined list of URLs.

FIG. **24** illustrates, by way of example, the second representation of the context-sensitive search form **25** presented to the user for the first embodiment. The context-sensitive search form **25** is displayed in web browser **41** as a Web page that includes display area **126** now used for displaying all or a portion of the refined list of URLs. Each URL in the refined list of URLs is preferably displayed as a hyperlink and includes some information from the corresponding document (e.g. title, a subset of text retrieved from the document). The context-sensitive search form **25** may also include context box **128** for displaying all or a portion of the refined list of contextual terms used to generate the refined list of URLs. If more than one contextual term remains in the context box **128**, the user may select from the remaining contextual terms to initiate a further search of the index **30** (FIG. **1**) to further refine the already refined list of URLs.

The combination of presenting the user with a list of data sources and a list of contextual terms associated with one or more data sources, and providing the user with a mechanism for selecting from the list of contextual terms so as to contextually refine the list of data sources, provides the user with a number of advantages. The search engine **24** automatically determines for the user the context within which the user's initial set of search criteria are used in the documents identified in the search results from the initial search. The user does not have to guess or try to predict the contexts within which the user's search terms are used in structured data sources, such as XML documents, identified in the search results. This automated technique for presenting the user with contextual information associated with the list of data sources retrieved from the initial search gives the user a mechanism for contextually filtering out data sources identified in the initial search. Thus, the user can quickly narrow the search results according to the context given to terms, phrases and other sets of data within structured data sources indexed within the index **30**.

Referring to FIG. **1**, the system **20** may have other aspects to further enhance functionality and usability. Each of the following aspects individually provides a beneficial enhancement and is an embodiment of the present invention.

In one variation, the index **30** maps contextual terms that it stores to an index of general (or normalized) contextual categories or generic contextual terms. In this variation, when contextual terms are retrieved in a search, their corresponding general contextual categories in the normalized index may be retrieved and displayed as a generic list of contextual terms or categories (or both) in a location of the context-sensitive search form **25**, or in place of the contextual terms displayed in the context box **128**. By selecting one of the generic contextual terms from the context-sensitive search form **25**, the user can then have the search engine **24** refine the initial search results by filtering out URLs referring to documents which do not contain either the generic contextual term selected or any of those contextual terms in the index **30** associated with the selected generic contextual term. For example, if the contextual terms "actor", "cameo", "stunt man", "performer" and "stage performer" are stored within the index **30** and are all associated with the generic contextual term "performer", then selection of this generic contextual term from a list of generic contextual terms presented to the user provides the basis for reducing the list of URLs to those having documents which contain any of the contextual terms associated with "performer". In a further variation, the search engine **24** is programmed to refine the list of URLs to only those URLs referring to documents identified in the index **30** as having one or more of the initial search terms marked by one or more contextual terms associated with the generic contextual term. In another alternative, the selection of a generic contextual term may be used not only to reduce the number of URLs referred to, but to potentially increase the number of URLs listed. In this latter case, the search engine **24** may be programmed, in response to the selection of a generic contextual term, to retrieve all URLs identified in the index **30** as referring to a structured document having at least one of the contextual terms aliased by the generic contextual term, provided that such contextual terms provide context to data that match the user's search criteria.

Referring to FIG. **1**, in the first embodiment, the user's machine does not communicate directly with the search engine **24**, but instead goes through the web computer server **27** and intermediary search application **26**. In one variation, the intermediary search application **26** may reside on the computer system **20** that supports the search engine **24**. In another alternative, client-side processing on the user's machine with, for example, style sheets or an applet such as a Java app, may be used to communicate directly with the search engine **24** without web server software or a web computer server, as illustrated in FIG. **27**. In this latter alternative, the user's browser serves as a search application interface to the search engine **24**, converting user-based search requests into raw XML requests which can be processed by the search engine **24**, and converting raw XML search results received from the search engine **24** into a format presentable to the user on the web browser.

Referring to FIG. **1**, in one variation, to assist the search engine **24** in performing a context sensitive search of the database **31**, the initial search form **23** includes a list of at least one contextual term stored within the index **30** in database **31**. This list provides the user with an indication of some or all of the contextual terms available within the index **30** to assist the user in formulating a context-based search request. The context-based search request can contain one or more contextual terms defining the context within which the search is to be performed on some or all of the other search terms that form part of the search request. In receiving the at least one contextual term with the context-based search request, the search engine **24** can then search the database **31** for references to documents and other data sources (for example, Web pages) having data elements such as character data associated with such contextual term(s) and return search results to the user machine that submitted the context-based search request.

In another alternative, if a search request received by the search engine **24** lacks any contextual terms used within the database **31**, the search engine **24** is programmed to compare the search term(s) submitted with the search request with contextual terms used by the database **31**. If the search engine **24** determines that any contextual terms within the database **31** provide context to any of the search terms, the search engine **24** then provides the user machine that submitted the search request with a list of contextual terms in the database **31** that are found to be associated with one or more of the search terms, along with instructions providing the user with the option to submit a context-based search request using one or more of the contextual terms provided. The user may then refine the search request and specify the context sensitive nature of the search request by selecting one or more of the contextual terms from the list of contextual terms retrieved by the search engine **24**. The refined search request (now context-based), once received by the search engine **24** may be used to conduct a context sensitive search for structured documents referenced by the database **31**.

In yet another alternative, the searching and retrieval of data described in the first embodiment may be performed on a relational database which associates data elements scanned from data sources with contextual terms from such data sources in the manner described in the first embodiment for index **30**.

Although this invention has been described with reference to illustrative and preferred embodiments of carrying out the invention, this description is not to be construed in a limiting sense. Various modifications of form, arrangement of parts, steps, details and order of operations of the embodiments illustrated, as well as other embodiments of the invention, will be apparent to persons skilled in the art upon reference to this description. It is therefore contemplated that the appended claims will cover such modifications and embodiments as fall within the true scope of the invention.

What is claimed is:

1. A computer-implemented method of retrieving data based on context, comprising:

(a) receiving a search criterion from a requesting party;

(b) searching a database for data sources which contain a data element that matches the search criterion;

(c) retrieving a set of structural components that provide context to the data element in any of the data sources; and

(d) transmitting the set of structural components and references to the data sources to the requesting party.

2. The method of Claim 1, further comprising restricting the data sources to a set of structured documents which contain a structural component selected from the set of structural components that provide context to the data element in the set of structured documents.

3. The method of Claim 2, further comprising transmitting references to the restricted set of structured documents to the requesting party.

4. The method of Claim 2, further comprising receiving from the requesting party an indicator identifying a structural component selected from the set of structural components transmitted to the requesting party.

5. The method of Claim 4, further comprising restricting the set of structured documents to structured documents which contain the structural component identified by the indicator.

6. The method of any of Claims 1 to 5, wherein the search criterion is received from a user machine.

7. The method of any of Claims 1 to 6, wherein the search criterion is used to search the database whose structure is hidden by a search application interface.

8. A computer-implemented method of retrieving data based on context, comprising:

(a) receiving a set of search criteria from a requesting party;

(b) retrieving from a database a set of references to data sources comprising structured documents which contain data elements matching the search criteria;

(c) generating a set of contextual terms identified in the database as providing context to at least one of the data elements in the structured documents; and

(d) transmitting the set of references and the set of contextual terms to the requesting party.

9. The method of Claim 8, further comprising restricting the set of references to references to structured documents which contain a contextual term selected from the set of contextual terms.

10. A computer-readable medium having stored instructions for use in the execution of the method of Claim 8.

11. A computer-implemented method of retrieving data, comprising:

(a) identifying a set of structural components based on one or more search criteria received from a requesting party;

(b) transmitting the set of structural components to the requesting party for selection;

(c) receiving a selected structural component from the set of structural components transmitted to the requesting party;

(d) retrieving references to structured documents that contain data associated with the selected structural component; and

(e) transmitting the references to the requesting party.

12. The method of Claim 11, further comprising identifying a set of references to structured documents that contain data marked by at least one of said set of structural components and transmitting said set of references to structured documents to the requesting party with the transmission of the set of structural components.

13. The method of Claim 11, wherein said set of structural components is identified as being a set of contexts associated with a set of structured documents.

14. The method of Claim 11, wherein said set of structural components is identified as being associated with a set of document structures.

15. A computer-readable medium having stored instructions for use in the execution of the method of Claim 11.

16. A computer system comprising:

(a) a processor;

(b) a network interface in communication with said processor for connection with a network; and

(c) memory in communication with said processor, said memory comprising computer-readable instructions adapting said processor to:

(i) receive a set of search criteria from a requesting party;

(ii) retrieve from a database a set of references to structured documents which contain data elements matching the search criteria;

(iii) generate a set of contextual terms identified in the database as providing context to the data elements in the structured documents; and

(iv) transmit the set of references and the set of contextual terms to the requesting party.

17. A system for retrieving context sensitive data, comprising:

(a) means for receiving a set of search criteria from a requesting party;

(b) means for retrieving from a database a set of references to data sources which contain data elements matching the search criteria;

(c) means for generating a set of contextual terms identified in the database as providing context to the data elements in at least one of the data sources; and

(d) means for transmitting the set of references and the set of contextual terms to the requesting party.

18. A computer-readable medium for providing instructions for directing a processing unit to retrieve data based on context, by:

(a) receiving a search criterion from a requesting party;

(b) searching a database for data sources which contain a data element that matches the search criterion;

(c) retrieving a set of structural components that provide context to the data element in any of the data sources; and

(d) transmitting the set of structural components and references to the data sources to the requesting party.

19. A computer-implemented method of indexing data into a database, the method comprising:

indexing a data source within the database comprising:

scanning the data source in search of structural components that provide context to any data elements within the data source;

retrieving the data elements and the structural components from the data source;

recording organizational data representing an organization of the data elements and the structural components within the data source; and

storing the organizational information within the database.

20. The method of Claim 19, further comprising keeping track of position information and nesting information for both data elements and structural components retrieved from the data source.

21. The method of Claim 19, wherein the data elements represent textual components of the data source, wherein recording further comprises recording position information and nesting information for textual components retrieved from the data source, and recording position information, nesting information and boundary information for the structural components.

22. The method of Claim 19, further comprising converting the data source to an XML document prior to indexing the data source, if the data source is not originally in XML format.

23. A computer-readable medium having stored instructions for use in the execution of the method of Claim 19.

24. A computer system comprising:

(a) a processor;

(b) a network interface in communication with said processor for connection with a network; and

(c) memory in communication with said processor, said memory comprising computer-readable instructions adapting said processor to perform the method of Claim 19:

25. A system for indexing data into a database, comprising:

(a) means for scanning a data source in search of structural components that provide context to any data elements within the data source;

(b) means for retrieving the data elements and the structural components from the data source;

(c) means for recording organizational information representing an organization of the data elements and the structural components within the data source; and

(d) means for storing the organizational information within the database.

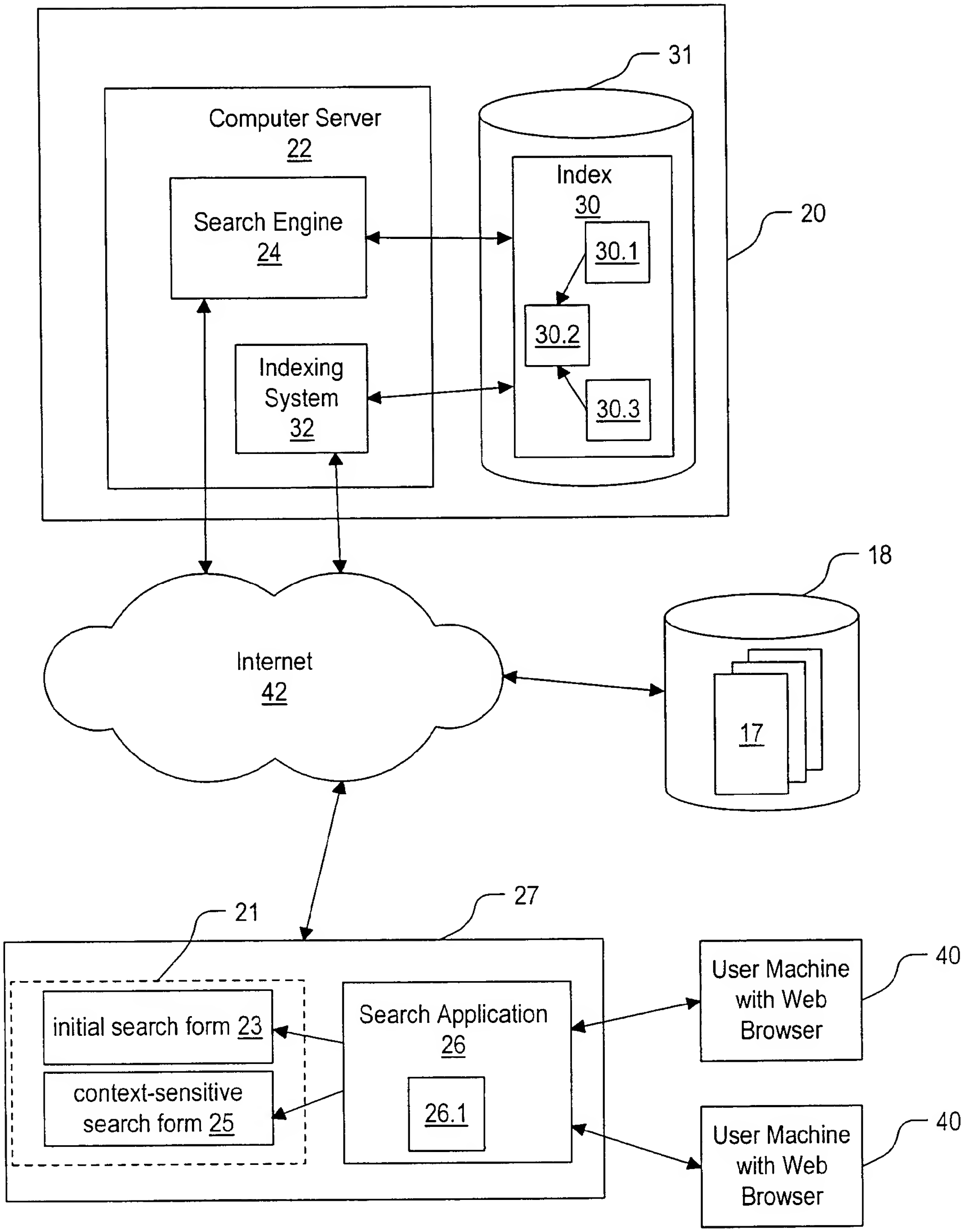

31
Computer Server 22
Search Engine 24
Index 30
30.1
30.2
30.3
20
Indexing System 32
18
Internet 42
17
21
27
40
initial search form 23
context-sensitive search form 25
Search Application 26
26.1
User Machine with Web Browser
40
User Machine with Web Browser

FIG. 1

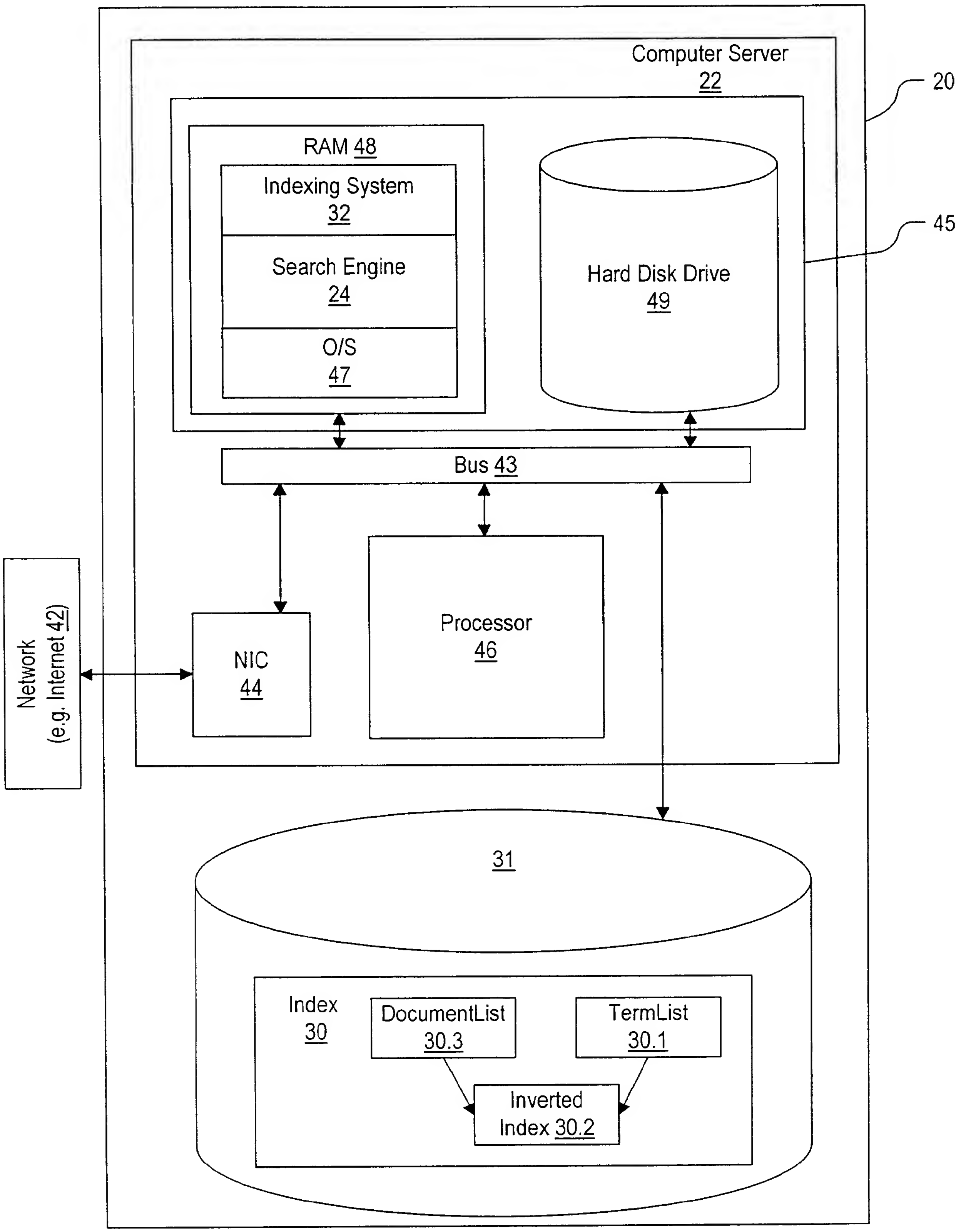

Computer Server
22
20
45
RAM 48
Indexing System
32
Search Engine
24
O/S
47
Hard Disk Drive
49
Bus 43
Processor
46
NIC
44
Network
(e.g. Internet 42)
31
Index
30
DocumentList
30.3
TermList
30.1
Inverted
Index 30.2

FIG. 2

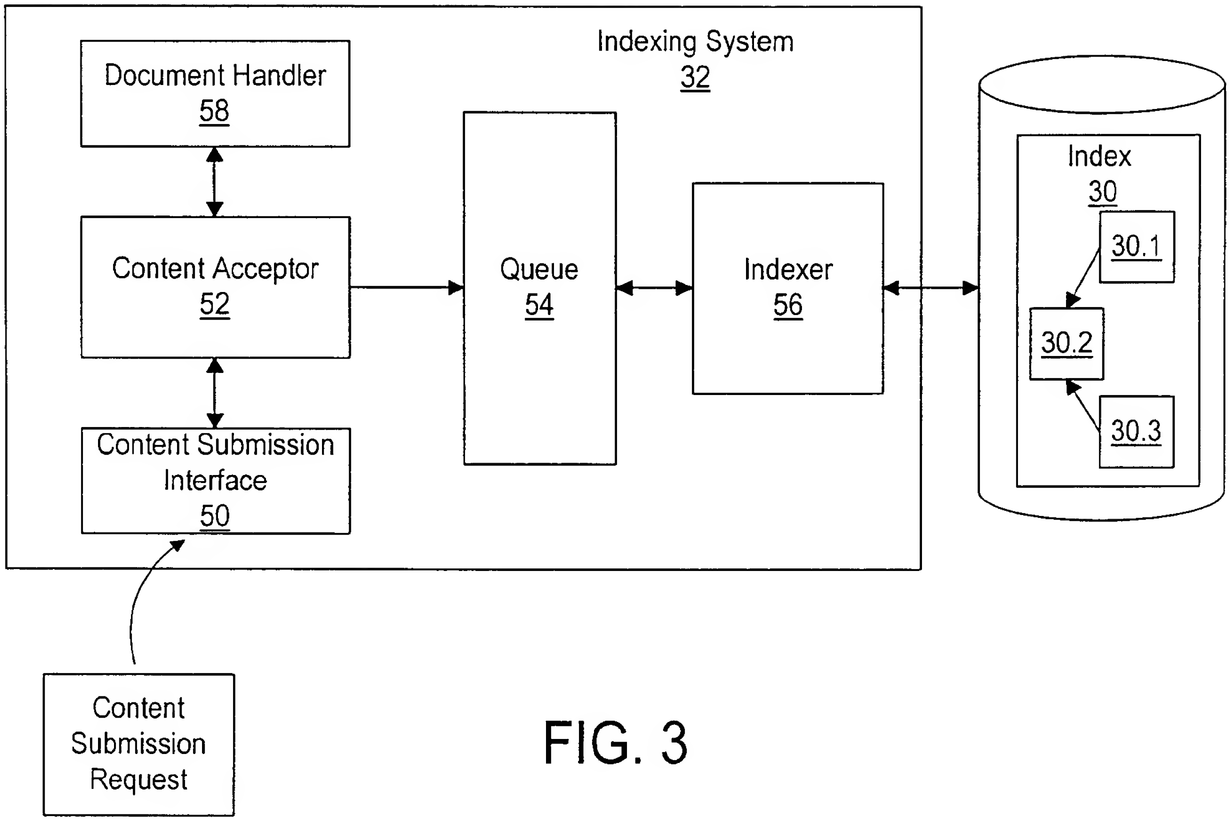


FIG. 3

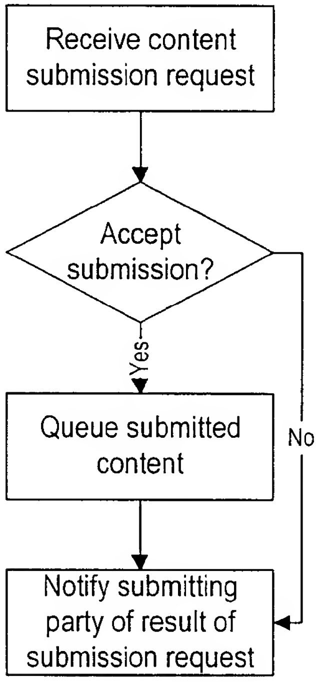

Receive content submission request
Accept submission?
Yes
No
Queue submitted content
Notify submitting party of result of submission request

FIG. 4

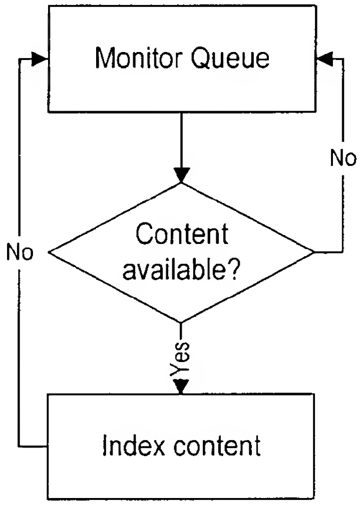
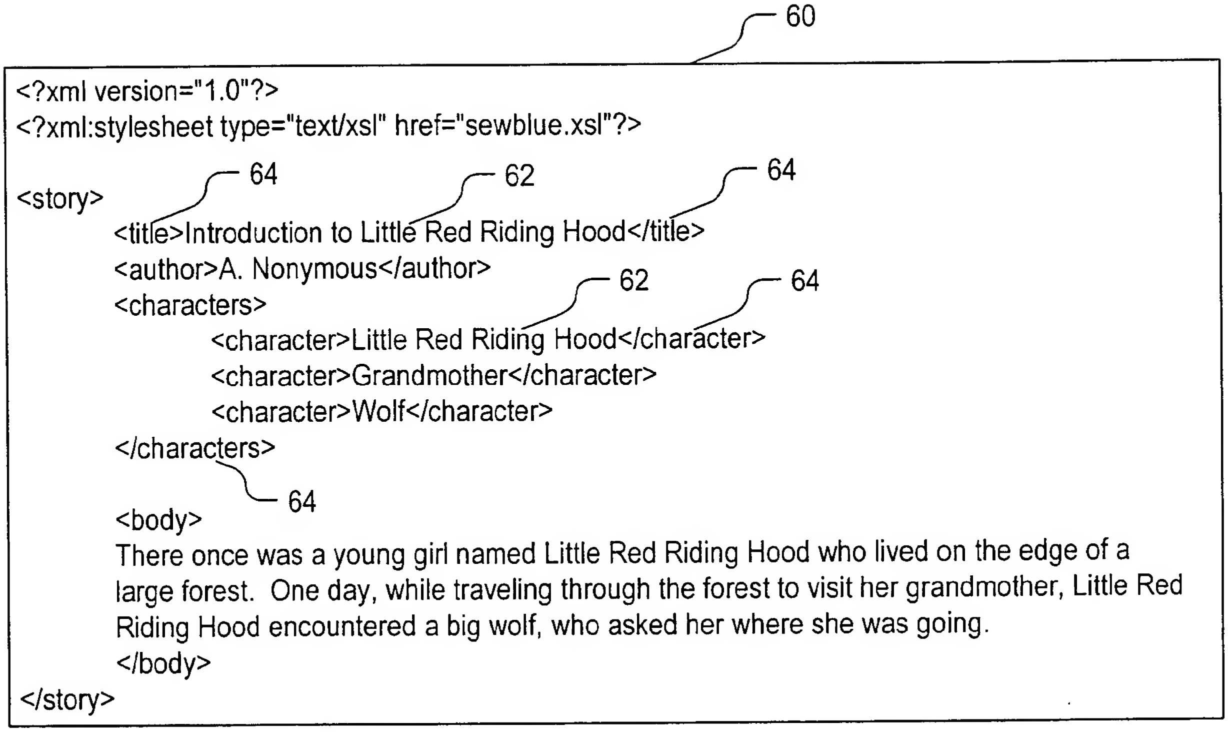
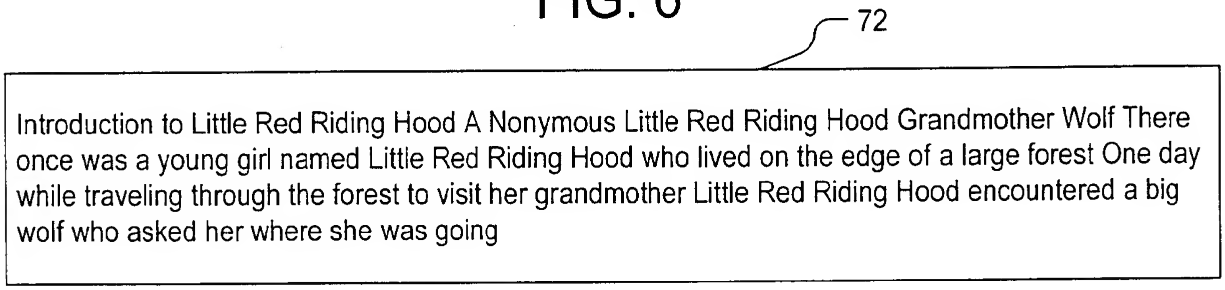


FIG. 5


60
64
62
64
62
64
64

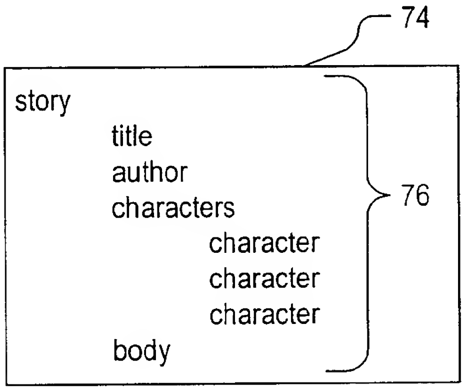
```
<?xml version="1.0"?>
<?xml:stylesheet type="text/xsl" href="sewblue.xsl"?>

<story>
        <title>Introduction to Little Red Riding Hood</title>
        <author>A. Nonymous</author>
        <characters>
                <character>Little Red Riding Hood</character>
                <character>Grandmother</character>
                <character>Wolf</character>
        </characters>
        <body>
        There once was a young girl named Little Red Riding Hood who lived on the edge of a
        large forest.  One day, while traveling through the forest to visit her grandmother, Little Red
        Riding Hood encountered a big wolf, who asked her where she was going.
        </body>
</story>
```

FIG. 6



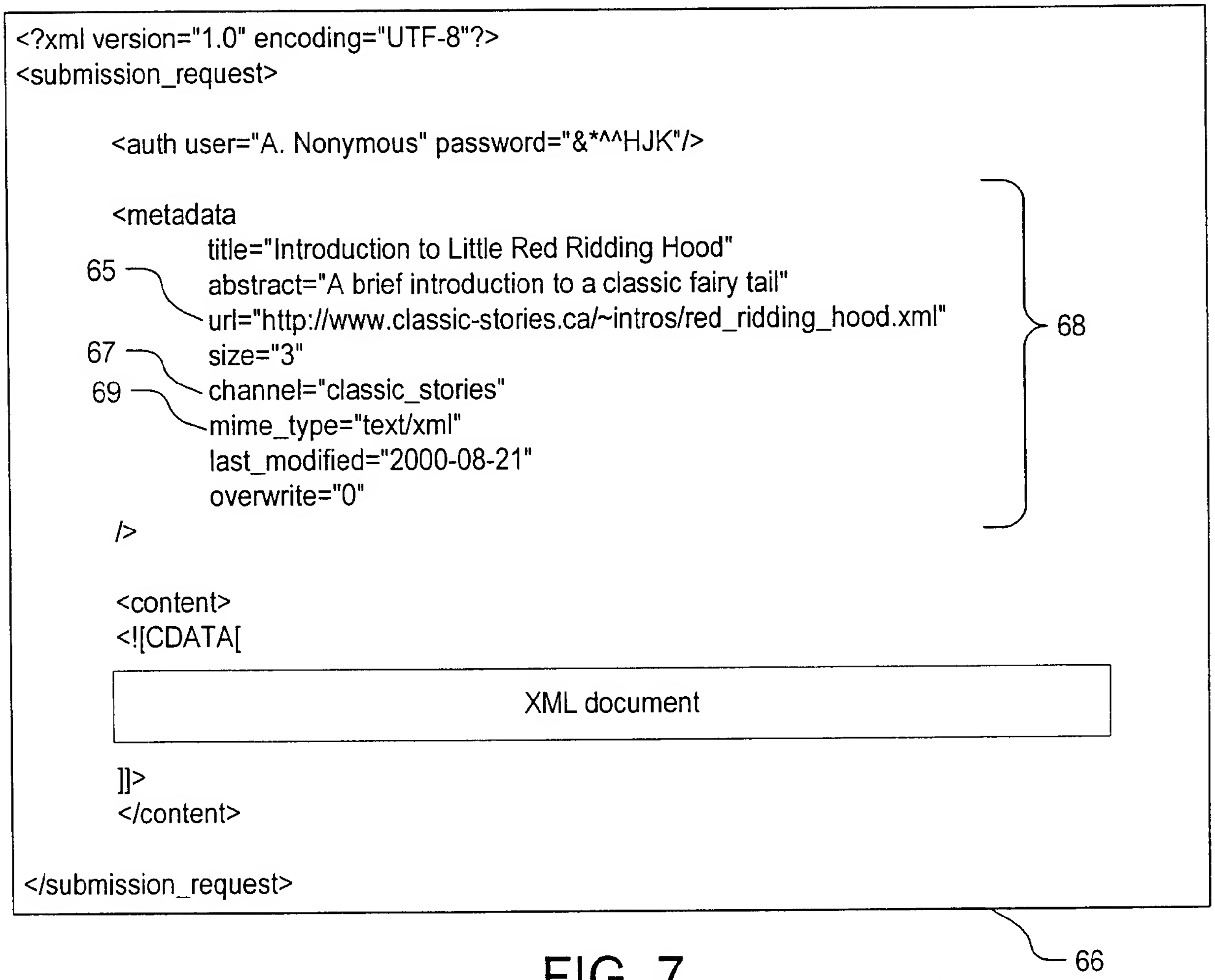
Introduction to Little Red Riding Hood A Nonymous Little Red Riding Hood Grandmother Wolf There once was a young girl named Little Red Riding Hood who lived on the edge of a large forest One day while traveling through the forest to visit her grandmother Little Red Riding Hood encountered a big wolf who asked her where she was going

FIG. 9



```
story
        title
        author
        characters
                character
                character
                character
        body
```

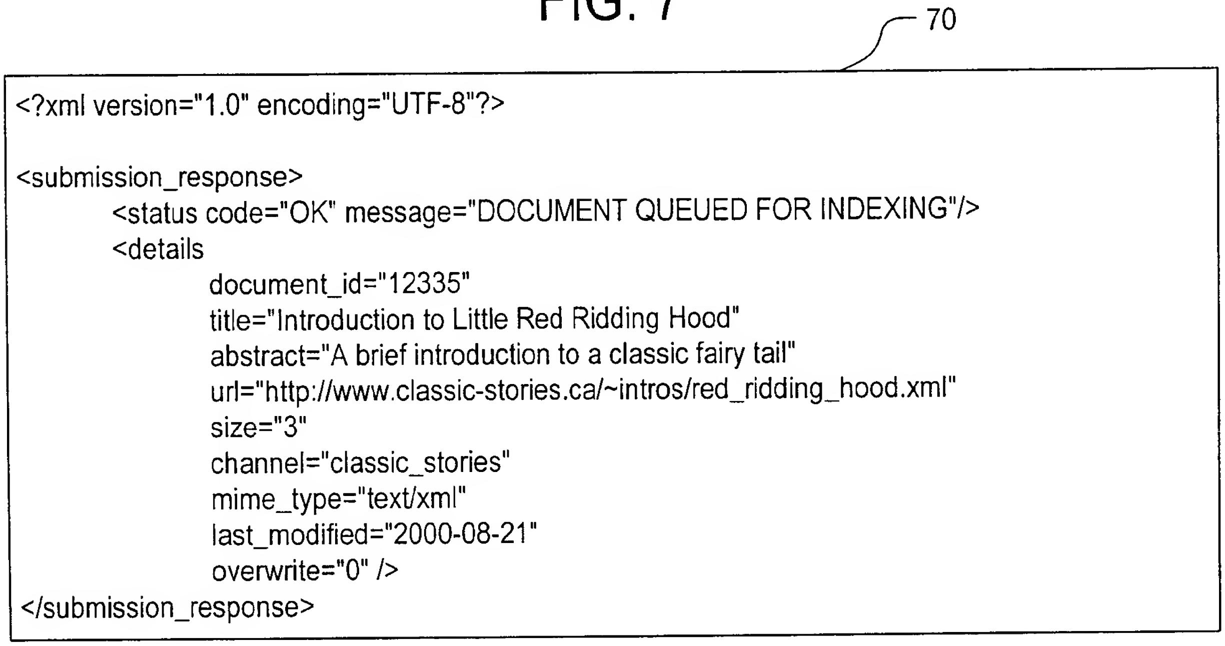
FIG. 10



```
<?xml version="1.0" encoding="UTF-8"?>
<submission_request>

        <auth user="A. Nonymous" password="&*^^HJK"/>

        <metadata
                title="Introduction to Little Red Ridding Hood"
                abstract="A brief introduction to a classic fairy tail"
                url="http://www.classic-stories.ca/~intros/red_ridding_hood.xml"
                size="3"
                channel="classic_stories"
                mime_type="text/xml"
                last_modified="2000-08-21"
                overwrite="0"
        />

        <content>
        <![CDATA[
                                XML document
        ]]>
        </content>

</submission_request>
```

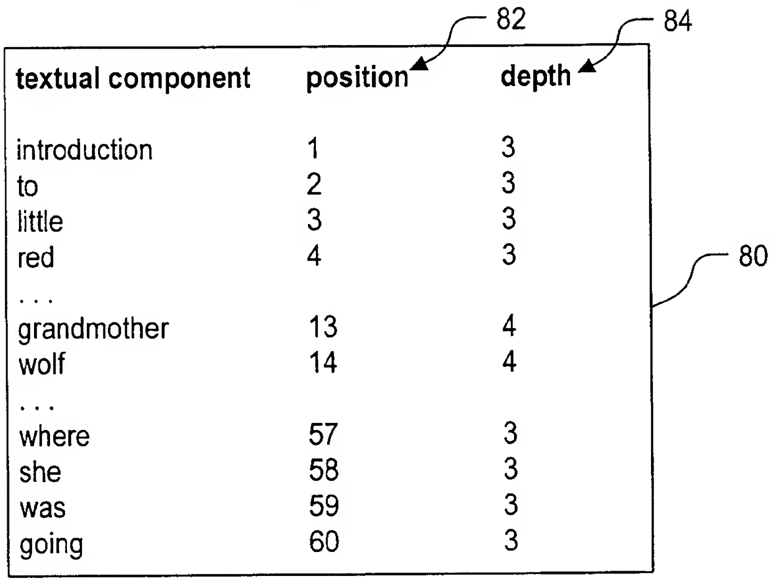
65
67
69
68
66

FIG. 7



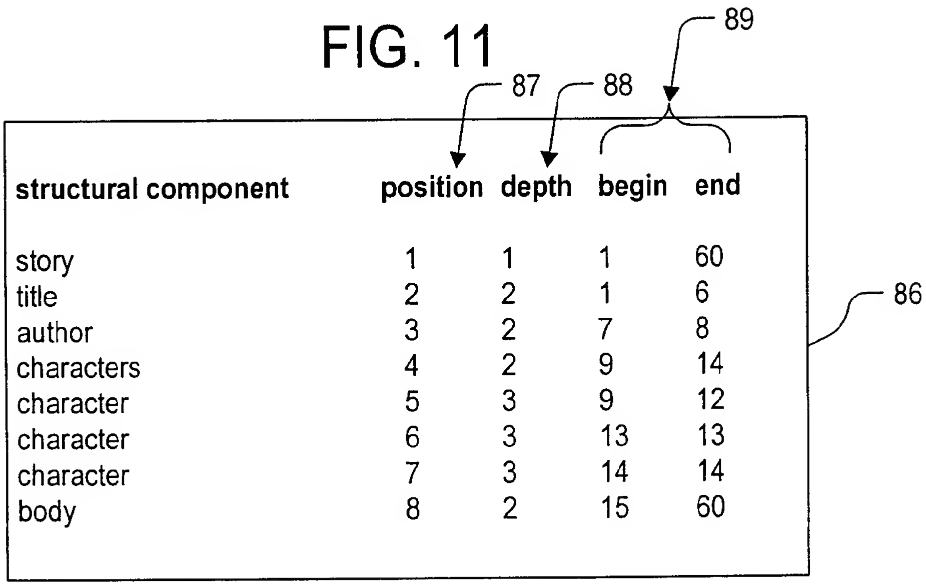
```
<?xml version="1.0" encoding="UTF-8"?>

<submission_response>
        <status code="OK" message="DOCUMENT QUEUED FOR INDEXING"/>
        <details
                document_id="12335"
                title="Introduction to Little Red Ridding Hood"
                abstract="A brief introduction to a classic fairy tail"
                url="http://www.classic-stories.ca/~intros/red_ridding_hood.xml"
                size="3"
                channel="classic_stories"
                mime_type="text/xml"
                last_modified="2000-08-21"
                overwrite="0" />
</submission_response>
```

FIG. 8



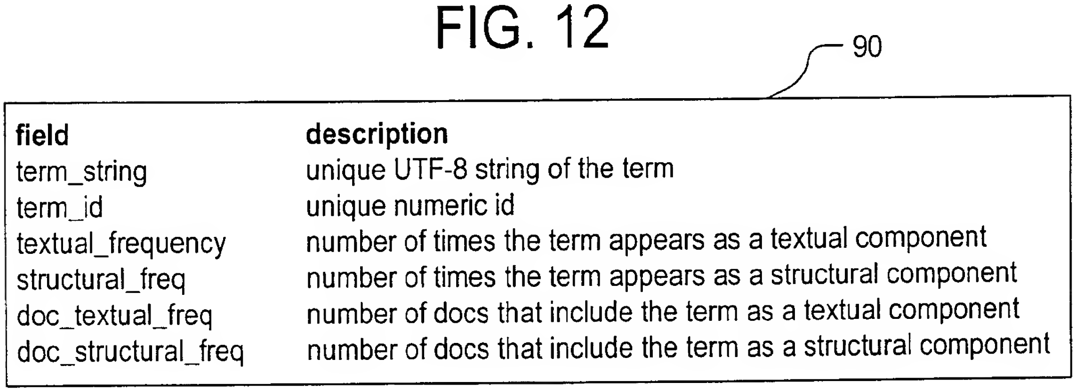
textual component	position	depth
introduction	1	3
to	2	3
little	3	3
red	4	3
. . .		
grandmother	13	4
wolf	14	4
. . .		
where	57	3
she	58	3
was	59	3
going	60	3

FIG. 11



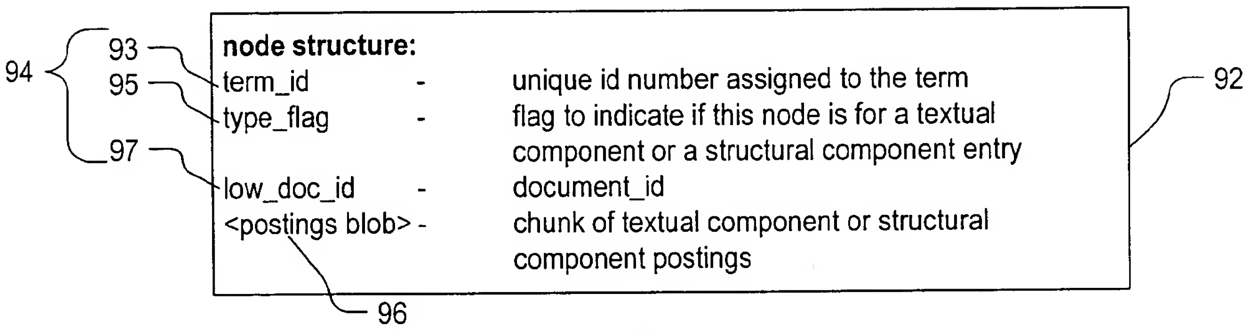
structural component	position	depth	begin	end
story	1	1	1	60
title	2	2	1	6
author	3	2	7	8
characters	4	2	9	14
character	5	3	9	12
character	6	3	13	13
character	7	3	14	14
body	8	2	15	60

FIG. 12



field	description
term_string	unique UTF-8 string of the term
term_id	unique numeric id
textual_frequency	number of times the term appears as a textual component
structural_freq	number of times the term appears as a structural component
doc_textual_freq	number of docs that include the term as a textual component
doc_structural_freq	number of docs that include the term as a structural component

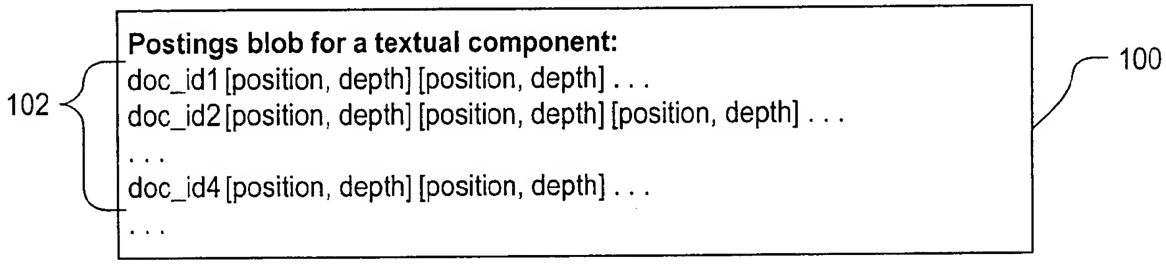
FIG. 13



node structure:

term_id	-	unique id number assigned to the term
type_flag	-	flag to indicate if this node is for a textual component or a structural component entry
low_doc_id	-	document_id
<postings blob>	-	chunk of textual component or structural component postings

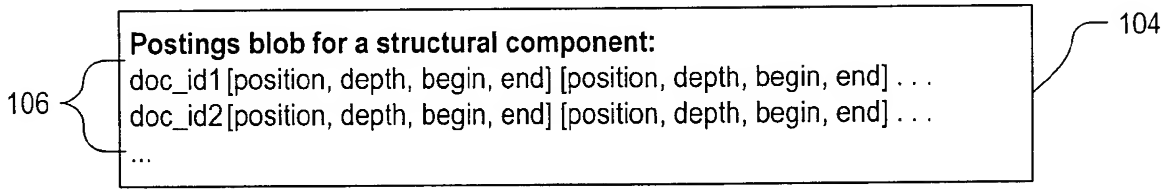
FIG. 14



Postings blob for a textual component:

doc_id1 [position, depth] [position, depth] . . .
doc_id2 [position, depth] [position, depth] [position, depth] . . .
. . .
doc_id4 [position, depth] [position, depth] . . .
. . .

FIG. 15



Postings blob for a structural component:

doc_id1 [position, depth, begin, end] [position, depth, begin, end] . . .
doc_id2 [position, depth, begin, end] [position, depth, begin, end] . . .
...

FIG. 16

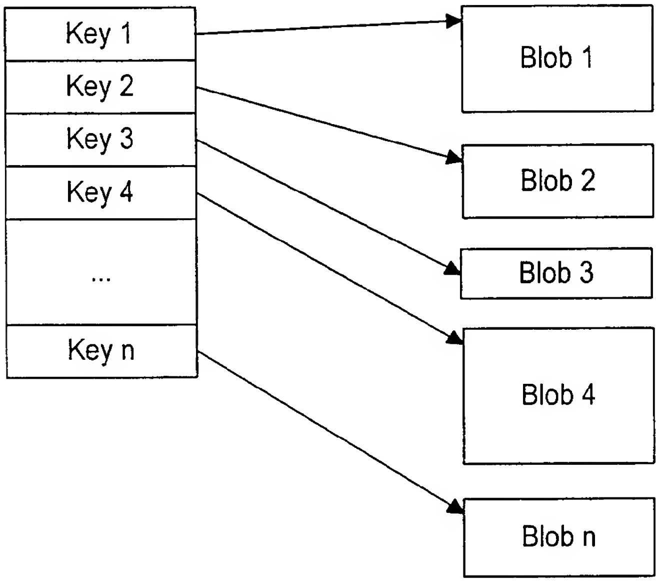

Key 1
Key 2
Key 3
Key 4
...
Key n
Blob 1
Blob 2
Blob 3
Blob 4
Blob n

FIG. 17

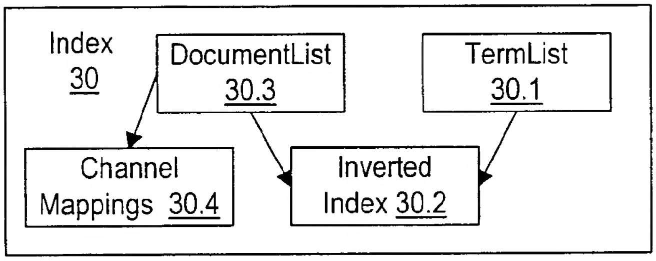

Index 30
DocumentList 30.3
TermList 30.1
Channel Mappings 30.4
Inverted Index 30.2

FIG. 18

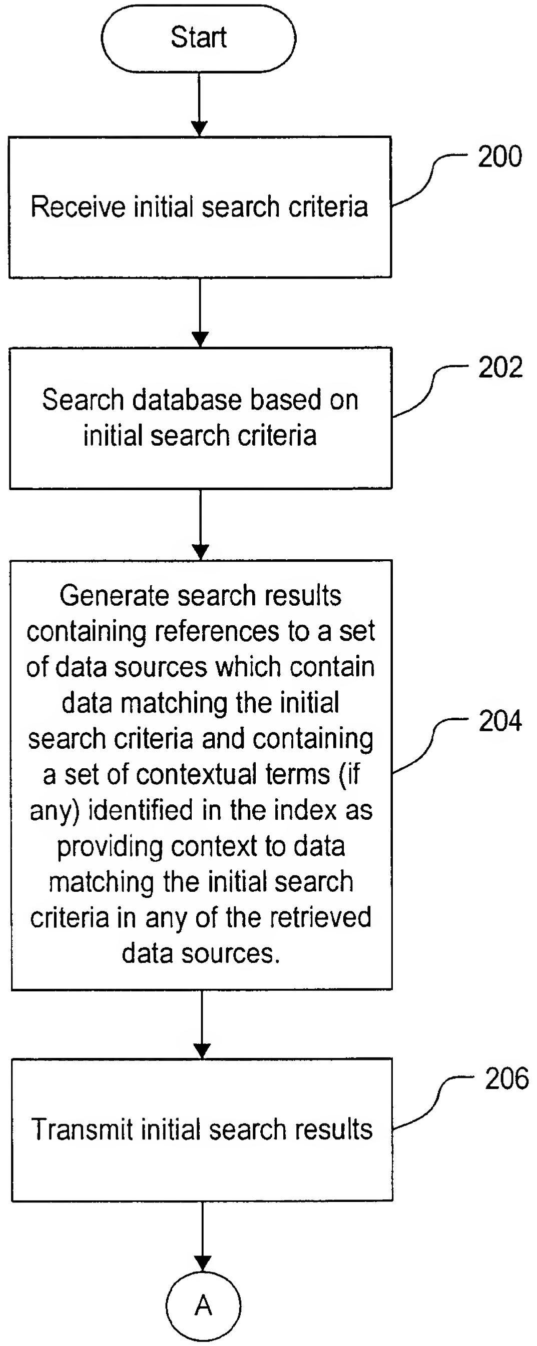

Start
Receive initial search criteria
200
Search database based on initial search criteria
202
Generate search results containing references to a set of data sources which contain data matching the initial search criteria and containing a set of contextual terms (if any) identified in the index as providing context to data matching the initial search criteria in any of the retrieved data sources.
204
Transmit initial search results
206
A

FIG. 19

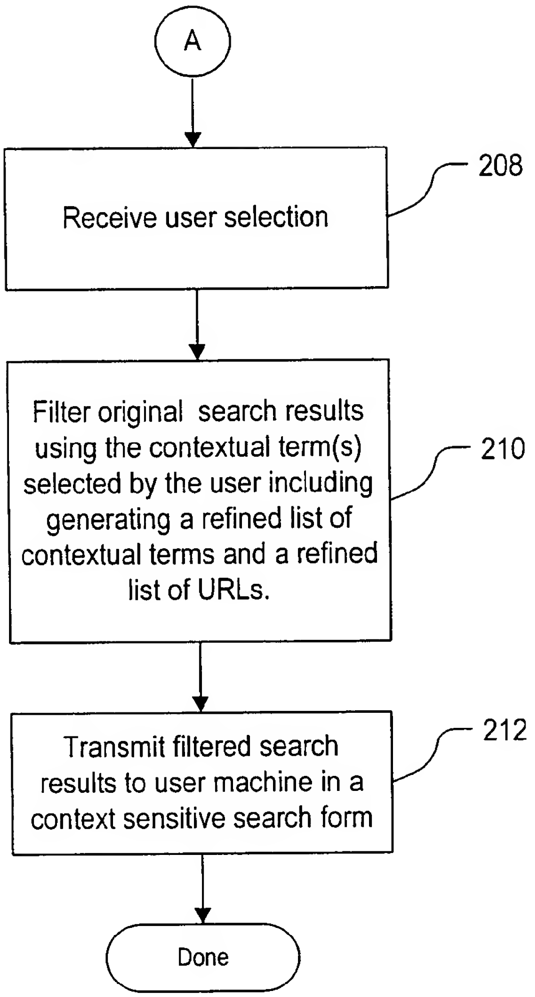

A
Receive user selection
208
Filter original search results using the contextual term(s) selected by the user including generating a refined list of contextual terms and a refined list of URLs.
210
Transmit filtered search results to user machine in a context sensitive search form
212
Done

FIG. 20

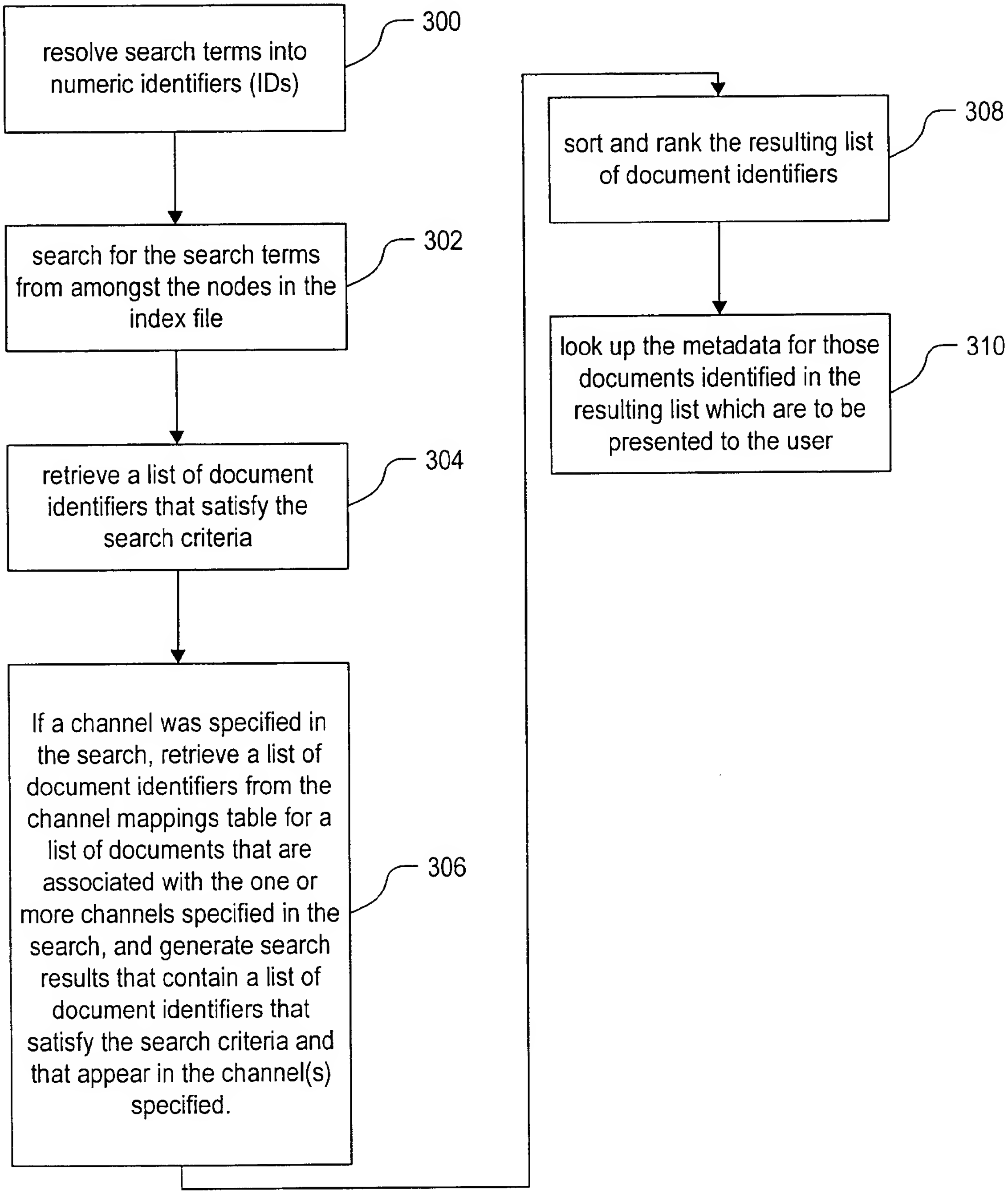

resolve search terms into numeric identifiers (IDs)
300
search for the search terms from amongst the nodes in the index file
302
retrieve a list of document identifiers that satisfy the search criteria
304
If a channel was specified in the search, retrieve a list of document identifiers from the channel mappings table for a list of documents that are associated with the one or more channels specified in the search, and generate search results that contain a list of document identifiers that satisfy the search criteria and that appear in the channel(s) specified.
306
sort and rank the resulting list of document identifiers
308
look up the metadata for those documents identified in the resulting list which are to be presented to the user
310

FIG. 20A

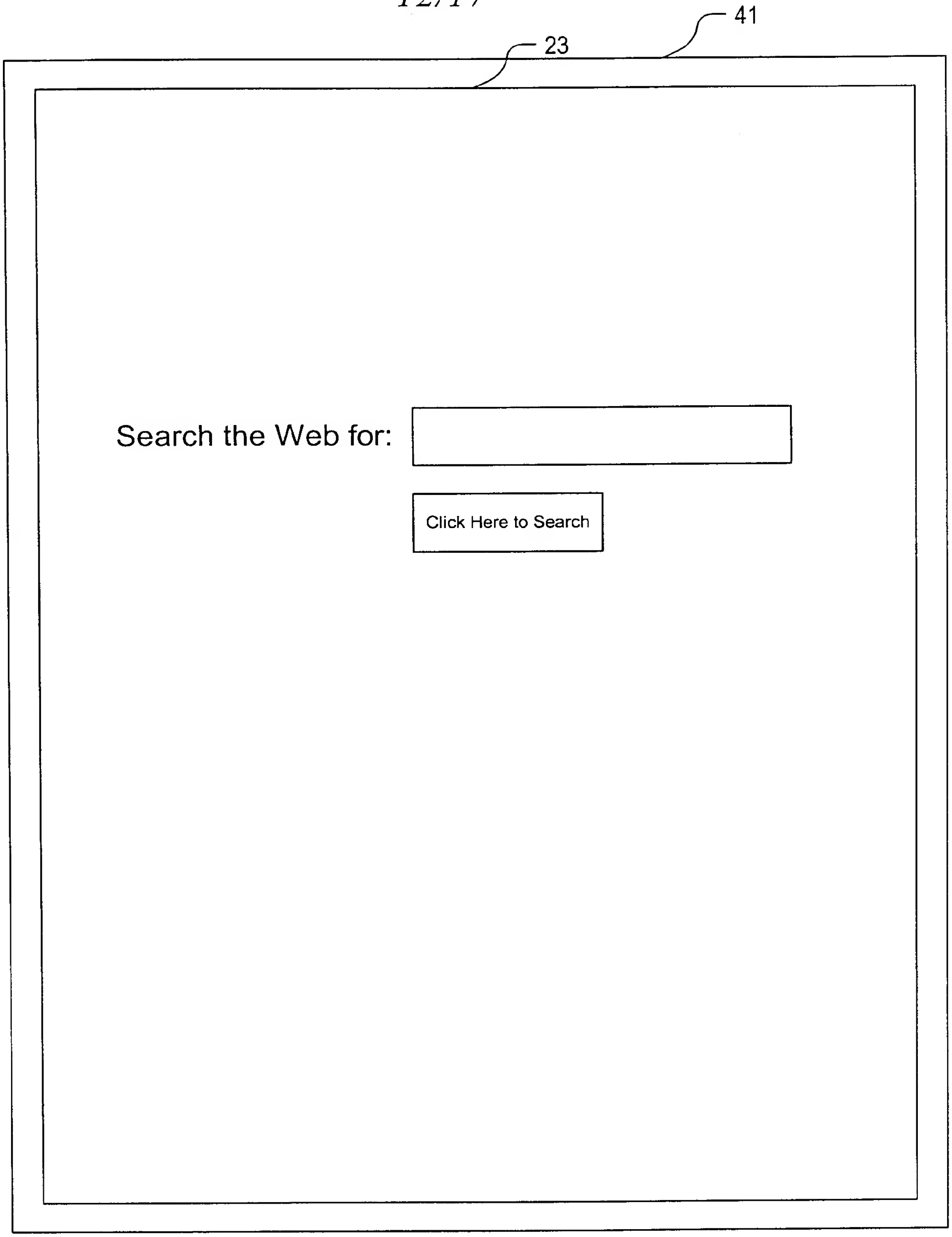


FIG. 22

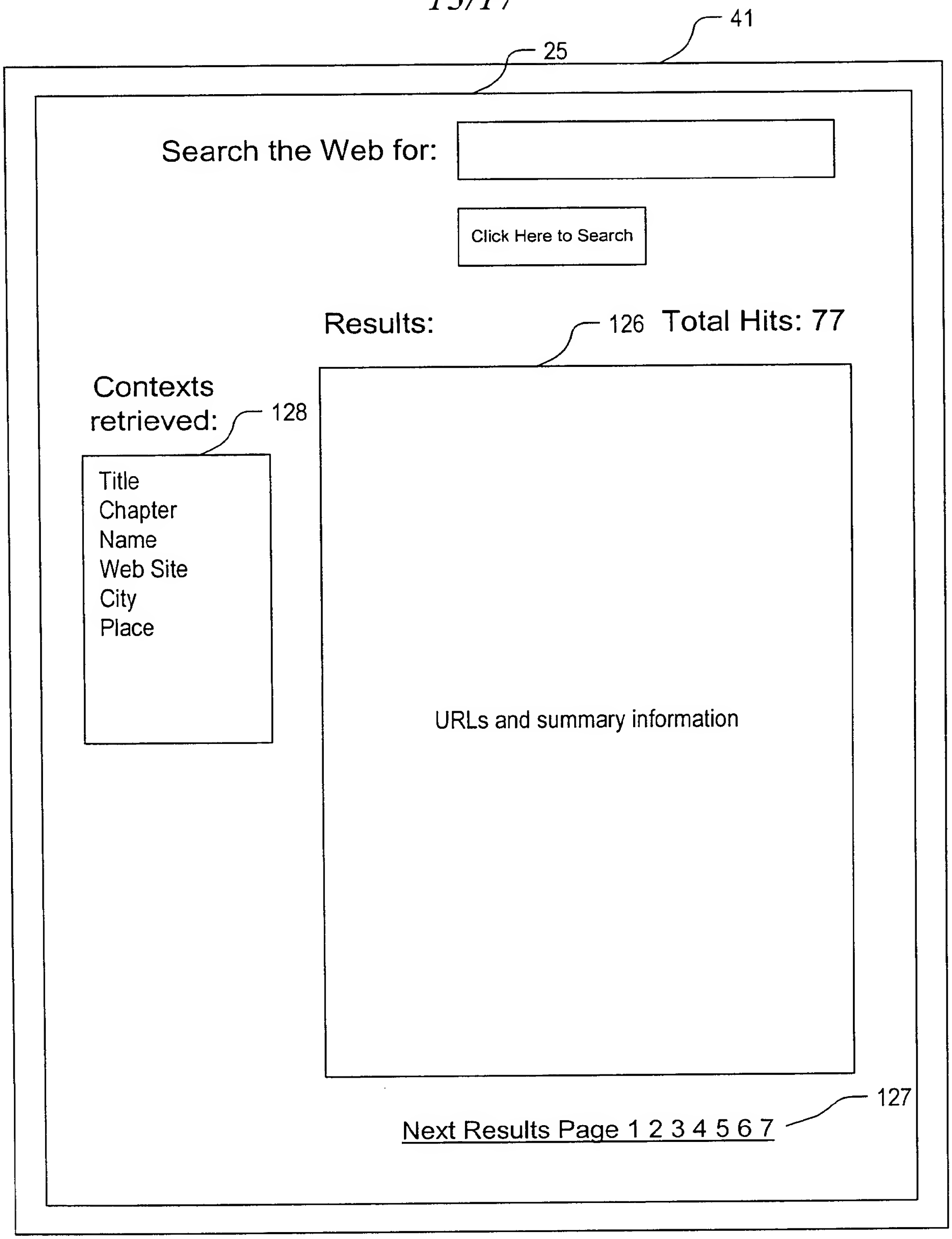

41
25
Search the Web for:
Click Here to Search
Results:
126
Total Hits: 77
Contexts retrieved:
128
Title
Chapter
Name
Web Site
City
Place
URLs and summary information
127
Next Results Page 1 2 3 4 5 6 7

FIG. 23

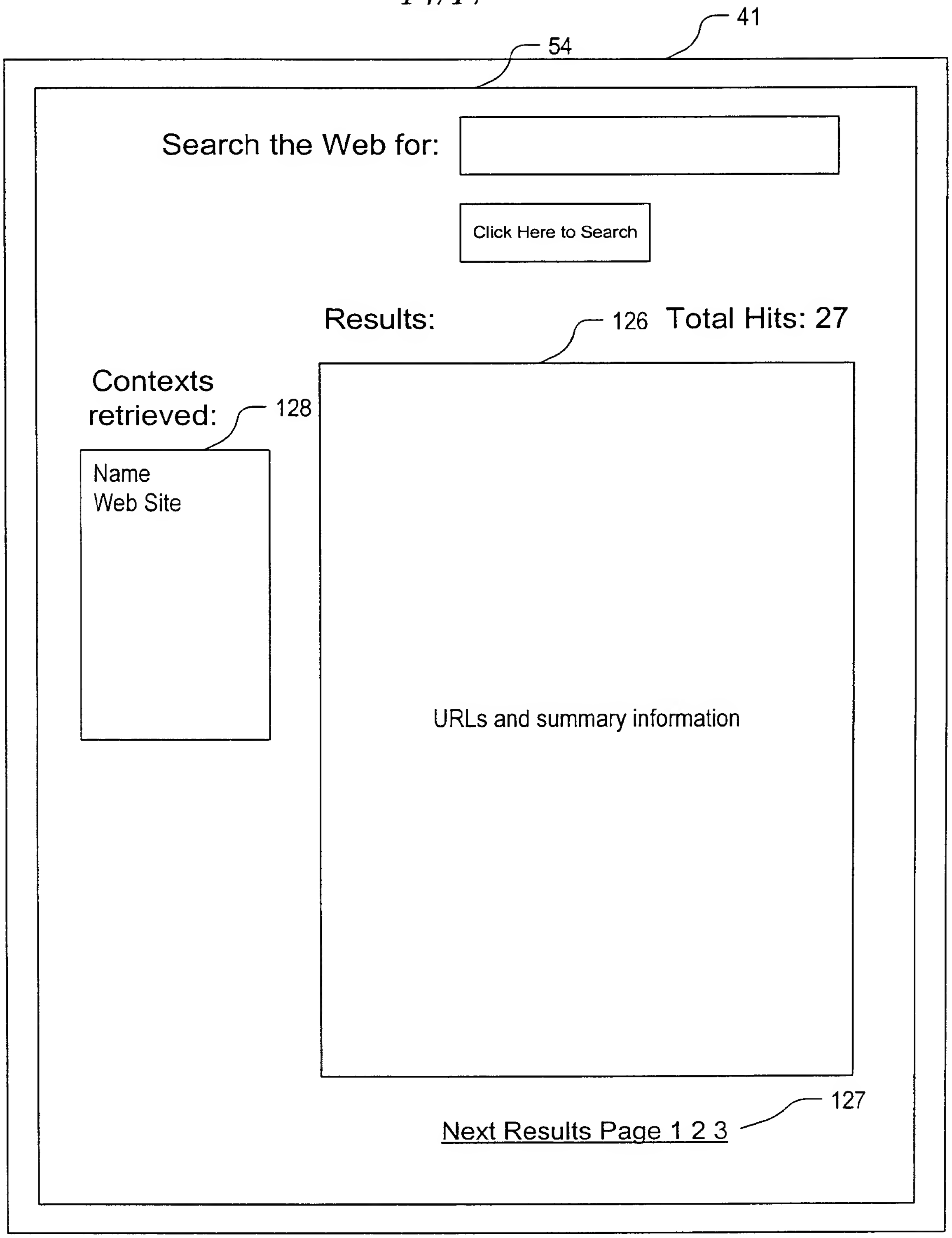

41
54
Search the Web for:
Click Here to Search
Results:
126
Total Hits: 27
Contexts retrieved:
128
Name
Web Site
URLs and summary information
Next Results Page 1 2 3
127

FIG. 24

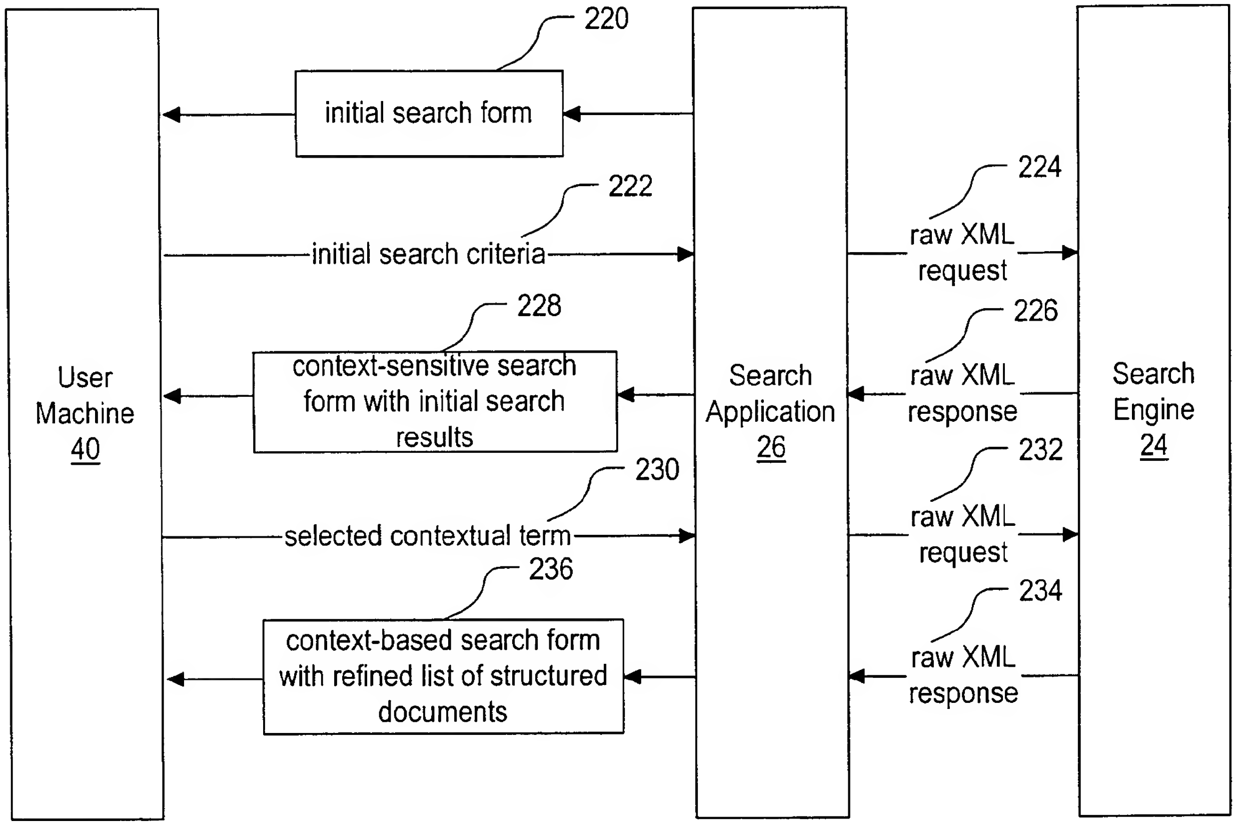
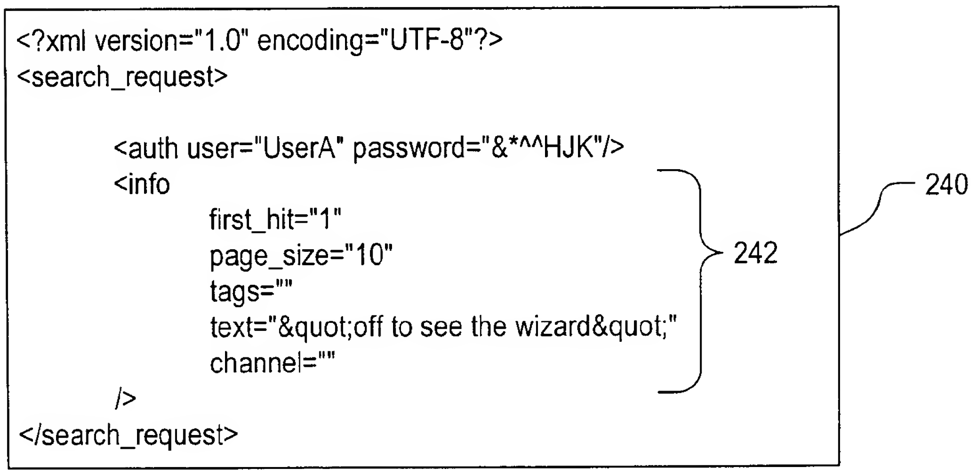
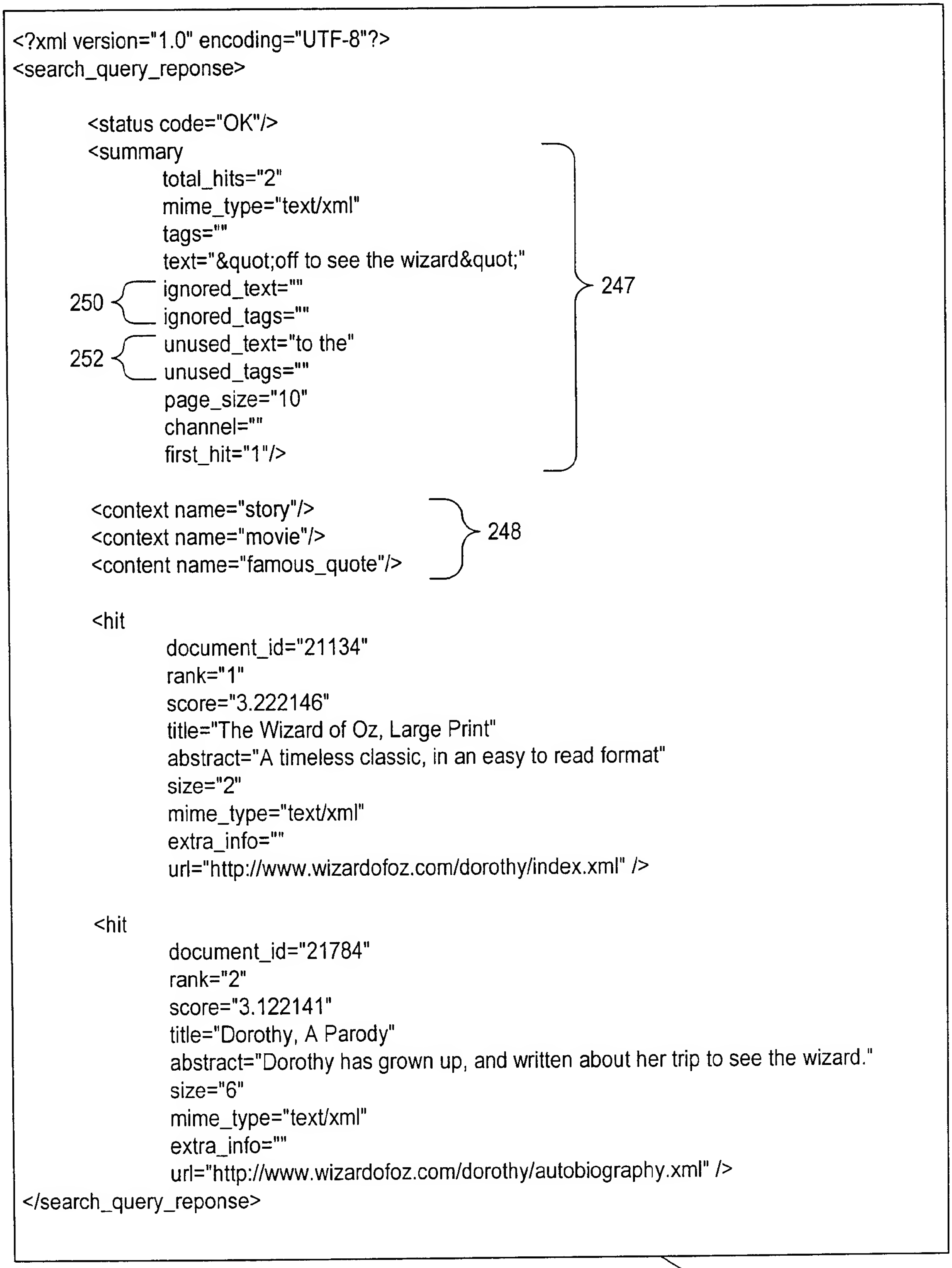


FIG. 21



```
<?xml version="1.0" encoding="UTF-8"?>
<search_request>

        <auth user="UserA" password="&*^^HJK"/>
        <info
                first_hit="1"
                page_size="10"
                tags=""
                text="&quot;off to see the wizard&quot;"
                channel=""
        />
</search_request>
```

FIG. 25



```
<?xml version="1.0" encoding="UTF-8"?>
<search_query_reponse>

        <status code="OK"/>
        <summary
                total_hits="2"
                mime_type="text/xml"
                tags=""
                text="&quot;off to see the wizard&quot;"
                ignored_text=""
                ignored_tags=""
                unused_text="to the"
                unused_tags=""
                page_size="10"
                channel=""
                first_hit="1"/>

        <context name="story"/>
        <context name="movie"/>
        <content name="famous_quote"/>

        <hit
                document_id="21134"
                rank="1"
                score="3.222146"
                title="The Wizard of Oz, Large Print"
                abstract="A timeless classic, in an easy to read format"
                size="2"
                mime_type="text/xml"
                extra_info=""
                url="http://www.wizardofoz.com/dorothy/index.xml" />

        <hit
                document_id="21784"
                rank="2"
                score="3.122141"
                title="Dorothy, A Parody"
                abstract="Dorothy has grown up, and written about her trip to see the wizard."
                size="6"
                mime_type="text/xml"
                extra_info=""
                url="http://www.wizardofoz.com/dorothy/autobiography.xml" />
</search_query_reponse>
```

FIG. 26

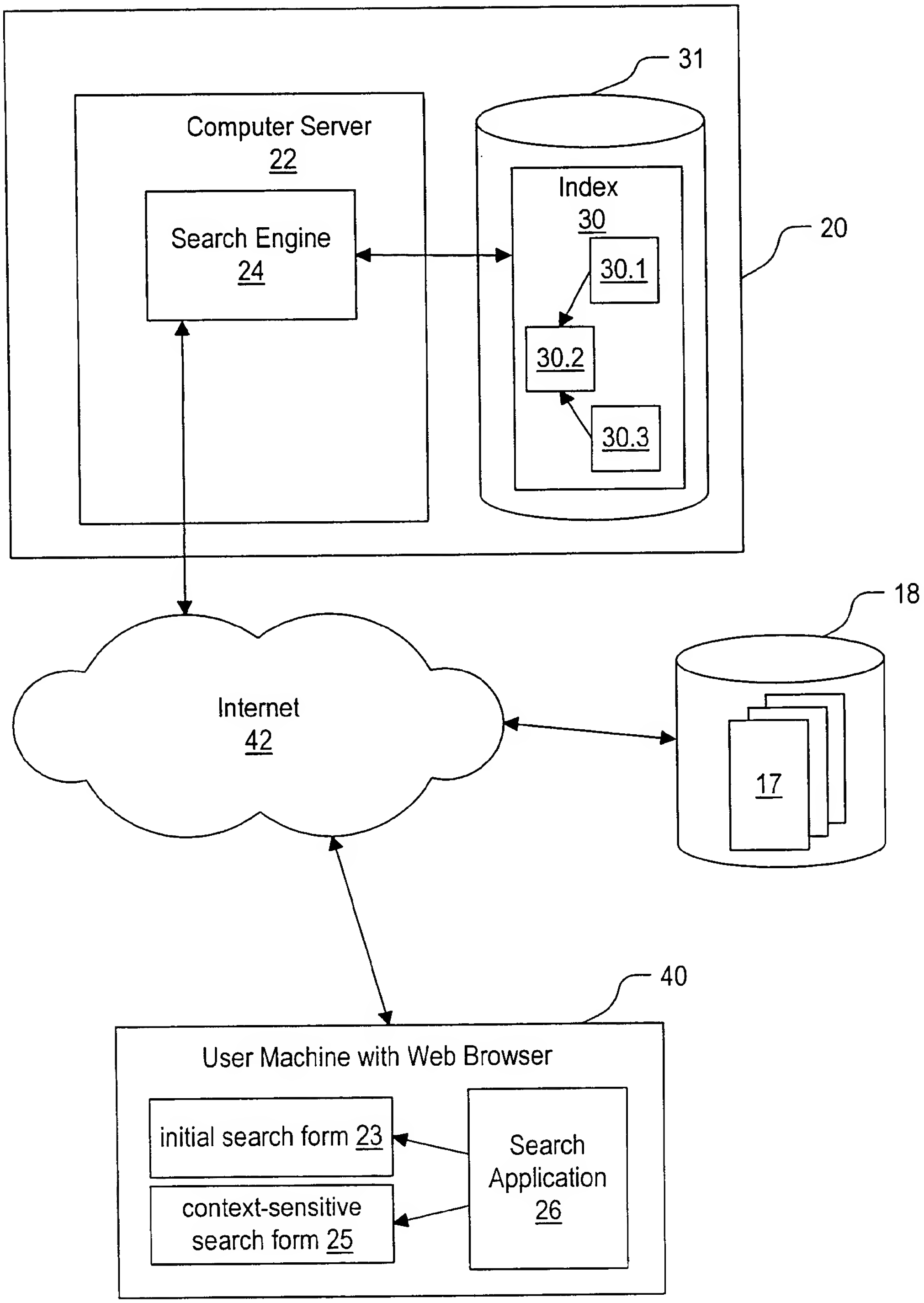

31
Computer Server
22
Search Engine
24
Index
30
30.1
30.2
30.3
20
18
Internet
42
17
40
User Machine with Web Browser
initial search form 23
context-sensitive search form 25
Search Application
26

FIG. 27